Lampada Dimmer a comando vocale

- IL TRANSISTOR: funzionamento digitale
- Telecomando infrarossi con ARDUINO
- Il Banana PI diventa SERVER

Fare Elettronica n. 360 - Ottobre 2015
SOMMARIO

L'Editoriale

di *Giovanni Di Maria*

Sistemi embedded e transistors

La tecnologia che offre oggi il mercato dell'elettronica è realmente stupefacente. Cosa ci sarà tra vent'anni? Sicuramente invenzioni e dispositivi futuristici. I progettisti di oggi possono avvalersi di sistemi embedded e di microcontrollori che, con poche righe di codice, generano prototipi dalle funzionalità estremamente complesse, difficilmente prevedibili con altri metodi. Tutto ciò, da un lato è molto interessante e utile e, anche dal punto di vista della qualità finale, la microprogrammazione è senz'altro da accettare. Ma dall'altro lato, essa fossilizza un po' il progettista a tralasciare la vera essenza dell'elettronica che si nasconde dietro. Chi è ancora in grado di calcolare e polarizzare un transistor? E' molto comodo oggi acquistare una busta di pasta già pronta, solo da riscaldare. Ma è altrettanto utile sapere preparare il piatto da zero, unendo a regola d'arte i vari ingredienti, per il miglior risultato, magari con un po' di fatica in più ma, sicuramente, con tanta soddisfazione.

Giovanni Di Maria

IL TRANSISTOR E LE SUE APPLICAZIONI
Parte terza: funzionamento digitale

di Vincenzo Sorce

Con questo articolo vedremo come sia fondamentale l'utilizzo del transistor, per realizzare qualsiasi tipo di circuito digitale. Fermo restando che nelle applicazioni pratiche si utilizzano i circuiti integrati, tuttavia l'uso dei singoli transistor risulta necessario nell'interfacciamento tra circuiti o per trasformare le uscite logiche in circuiti di potenza. Vedremo inoltre il loro uso per pilotare i relè.

Tutto quello di cui si è discusso nella prima e seconda parte c'entra poco o nulla rispetto a quanto diremo adesso. Precedentemente abbiamo parlato di punto di riposo e di zona attiva del transistor. Abbiamo visto che la tensione ai capi del transistor è sensibilmente alta dato che il segnale da amplificare deve interessare sia la parte destra che la parte sinistra del punto di riposo. La prima differenza fondamentale tra il suo utilizzo analogico e quello digitale è che la sua tensione ai capi del collettore ed emettitore deve essere quasi nulla. Ciò ha come conseguenza immediata che la potenza dissipata dovrà essere bassa, non ponendosi, così, io problema della dissipazione termica. E' fondamentale, per la nostra trattazione, che il componente si comporti come un interruttore. A tale scopo osserviamo la figura 1. Supponiamo di voler alimentare la resistenza R prima con un interruttore comune e poi con un interruttore digitale quale può essere, per esempio, l'uscita di un microcontrollore che dà il comando logico 0/1 con una tensione da 0 a un certo valore, in volt. Prima di tutto dobbiamo scegliere il transistor. Come si è potuto constatare, si sono fatti esempi sempre col transistore NPN connesso ad emettitore comune. Infatti, i transistors PNP non si producono più perché la loro realizzazione è costosa e le caratteristiche tecniche son inferiori rispetto a quelli con giunzione NPN. Riprendendo la nostra trattazione possiamo senz'altro affermare che per le applicazioni comuni, non richiedenti tensioni correnti e

Figura 1

frequenze particolarmente elevate, è molto impiegato il transistor BC547C. E' bene sottolineare che le caratteristiche di cui alla figura 1 sono puramente indicative e non corrispondono alle caratteristiche del transistor citato. La grandezza fondamentale da prendere in esame è la corrente che deve attraversare la resistenza di carico R, che nel nostro caso è 10mA. Se noi tracciamo, sul piano delle caratteristiche di uscita del transistor, una retta partendo dai 10mA dell'asse delle ordinate, e procedendo parallelamente all'asse delle ascisse, intersecheremo tutte le curve rappresentate per diversi valori di Ib. E' subito evidente che più elevato sarà il valore di Ib scelto e più bassa sarà la tensione ai capi del transistor. Dalla citata figura si riscontra subito che se scegliamo una Ib=0,1mA, alla quale corrisponde:

hfe=Ic/Ib=10mA/0,1mA=100

Si ha una tensione ai capi del transistore vicino all'unità. Se invece scegliamo una Ib=0,5mA si avrà una tensione ai capi del transistore prossimo allo zero e un hfe:

hfe=Ic/Ib=10mA/0,5mA=20

A questo punto dobbiamo soffermarci su alcune importanti considerazioni:

1) Più basso è l'hfe da noi imposto, più bassa sarà la tensione ai capi del transistor;
2) Normalmente i transistor hanno un hfe non inferiore a 100 (sono esclusi quelli di potenza) e che questo è il valore massimo che si sceglie per utilizzare il transistor come componente digitale;
3) La tensione ai capi del transistor in commutazione in generale è pari a 0,2 V;
4) La tensione ai capi della resistenza è opposta a quella applicata al circuito d'ingresso.

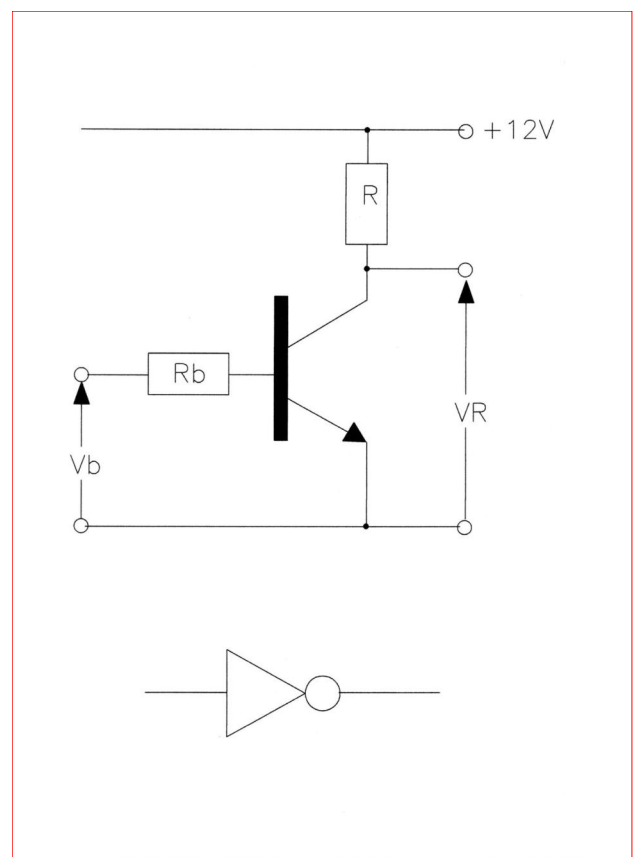



Figura 2

CIRCUITI LOGICI CON TRANSISTORS

PORTA NOT (Figura 2)

Se la tensione d'ingresso Vb è al livello logico 0, il transistor non conduce, perciò è un circuito aperto. Se ne deduce che la tensione VR si trova al livello logico 1, cioè a +12V. Se, invece, la Vb è al livello logico 1, poniamo a +5V, se la resistenza Rb ha il valore congruo, si avrà la conduzione come interruttore del transistore e, di conseguenza, la VR sarà 0,2 V corrispondente allo 0 logico. Facciamo un po' di conti: se R=1k quanto dovrà essere RB se Vb=5V ? La corrente che deve circolare sulla resistenza sarà :

Ic=(+12V-0,2V)/1000=11,8mA

Se imponiamo:

hfe=Ic/Ib=50

si avrà:

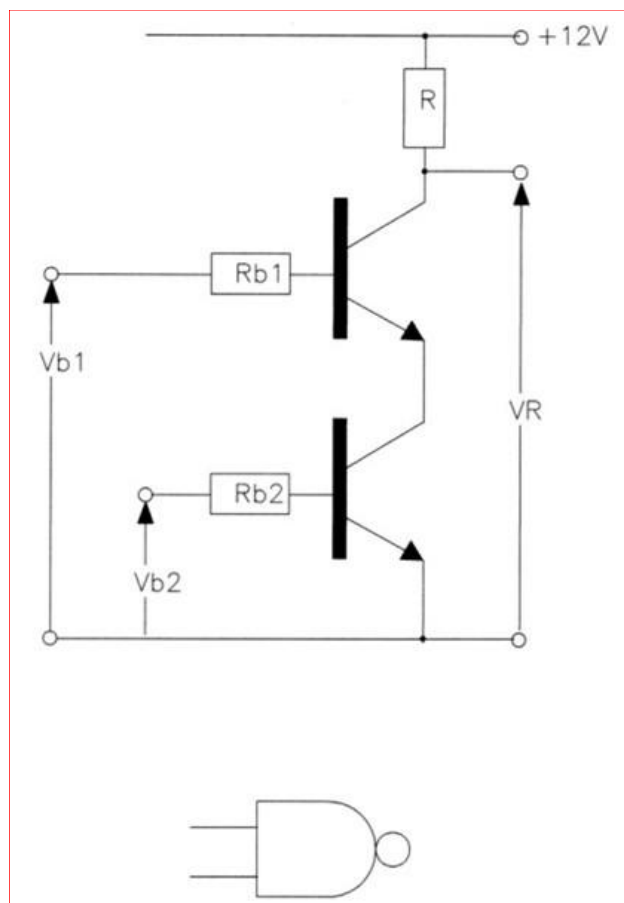



Figura 3

Ib=Ic/50=11,8mA/50=0,24mA

trascurando la caduta di tensione tra base ed emettitore potremo scrivere:

Rb=Vb/Ib=+5V/0,24mA=20K

Ovviamente sceglieremo il valore commerciale 18K.

PORTA NAND (Figura 3)

Se Vb1=0 e Vb2=0 i due transistors sono interdetti e la VR=1. Se uno dei due ingressi è 0 il corrispondente transistor è aperto e la VR=1. Se entrambi i due ingressi sono a livello logico 1 entrambi i transistori sono in conduzione e la VR è 0. Bisogna però notare che in tal caso si sommano le tensioni di collettore dei due transistors in conduzione e il livello 0 teorico in questo caso sarà 0,4V effettivo. Da ciò si evince che aumentando il numero dei transistors si potrebbero avere dei problemi.

PORTA NOR (Figura 4)

Quando le tre tensioni d'ingresso sono allo stato logico 0 si ha VR=1, mentre quando una delle tre è pari al livello logico 1 porta il transistor in conduzione e l'uscita sarà VR=0.

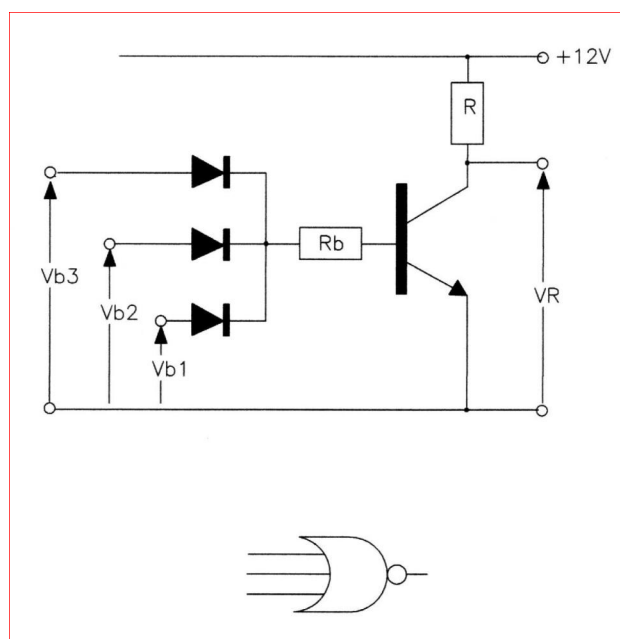



Figura 4

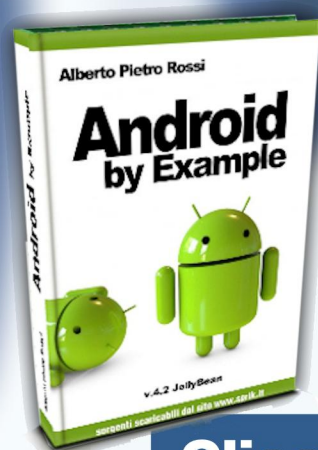
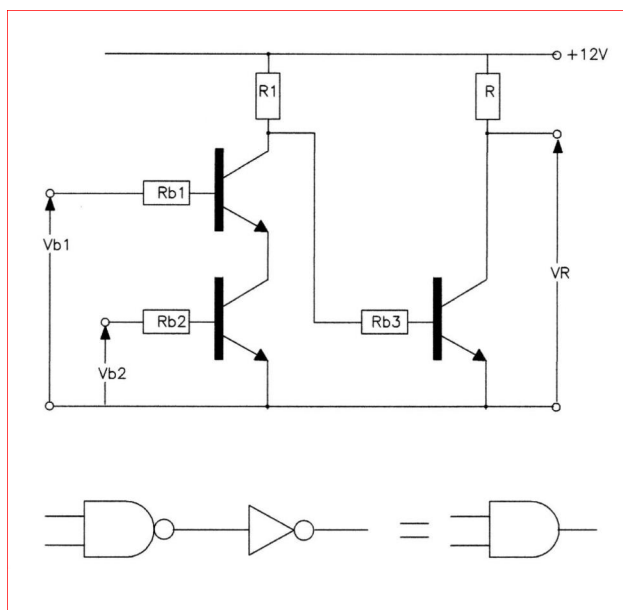



Figura 5

I tre diodi sono necessari per evitare che ci sia una reciproca influenza fra i circuiti d'ingresso.

PORTA AND (Figura 5)

In figura è mostrato come negare il NAND che abbiamo trattato. Nello stesso modo si può procedere con gli altri circuiti logici negati.

I TRANSISTORS NELL'INTERFACCIAMENTO CON I RELE'

Una delle applicazioni più interessanti del transistor è quella che lo vede come interfaccia tra una uscita logica e una uscita di potenza.

Come vedremo più in là le uscite dei circuiti logici integrati sono in grado di pilotare delle piccole potenze, come per esempio i led, ma non sono in grado di pilotare circuiti che presentano un carico elevato. Come abbiamo già visto, nell'utilizzo del transistor nel funzionamento da interruttore è necessario che si abbia un Hfe=Ic/Ib>=100. Supponiamo di dover pilotare (cioè accendere o spegnere) una stufa di 1000Watt in c.a. (corrente alternata a 230V. Per risolvere tale problema abbiamo la necessità di utilizzare, per esempio,

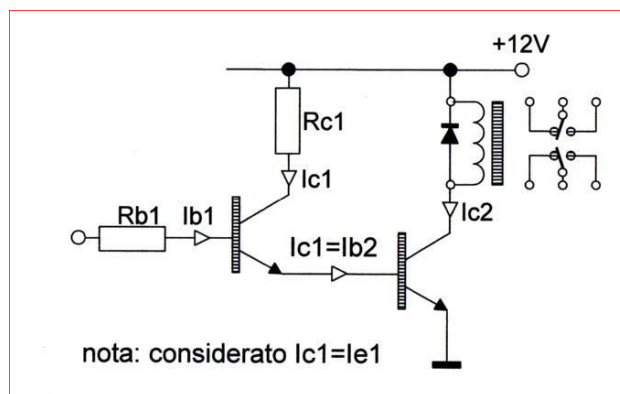



Figura 6

un relè che abbia le seguenti caratteristiche:

-Contatti di uscita 230V – 10 A;
-Bobina di alimentazione 12V c.c. (corrente continua) – 50 mA.

Se l'integrato logico ha una uscita con tensione 5V e corrente massima 5mA ci accorgiamo subito che è sufficiente un solo transistor, per esempio il BC547C che abbiamo già trattato, per ottenere che lo stesso funzioni come un interruttore. Naturalmente dobbiamo calcolarci il valore della resistenza di base. Come già visto la formula è la seguente:

RB=5V/5mA=1 kOhm

E' bene sottolineare che l'utilizzo del diodo in parallelo alla bobina del relè è di fondamentale importanza. Infatti all'apertura del transistor (transistor interdetto, cioè che non conduce), alla tensione della batteria si aggiunge la tensione ai capi della bobina di alimentazione del relè, che può portare alla distruzione del transistor.

Connessione Darlinghton

E' interessante considerare la connessione Darlinghton fra due transistors. Questo tipo di connessione è riportata in Figura 6. Il risultato di questa connessione è il seguente:

hfe1=Ic1/Ib1

da cui:

Ib1=Ic1/hfe1

hfe2=Ic2/Ib2

da cui:

Ic2=hfe2 x Ib2=hfe2 x Ic1

e considerando hfe complessivo:

hfe=Ic2/ib1=
hfe2 x Ic1/Ic1/hfe1
=hfe2 x hf1

Se, così come visto, se hfe1 e hf2 sono entrambi pari a 100, l'hfe complessivo sarà 10.000. Ciò vuol dire che per pilotare il relè dell'esempio di cui sopra basterà una corrente:

Ib1=50mA/10000=5uA

I CIRCUITI INTEGRATI LOGICI

Ovviamente la trattazione sin qui illustrata ha uno scopo essenzialmente didattico. In realtà raramente si utilizzano circuiti logici con componenti discreti, dato che esistono circuiti integrati che mettono a disposizione del progettista una miriade di porte And, Nand, Or,Nor, Not e così via. Come se ciò non bastasse con l'avvento dei microcontrollori è possibile realizzare complesse funzioni logiche. Non bisogna, però dimenticare che i principi su cui si basano sono quelli visti sopra.

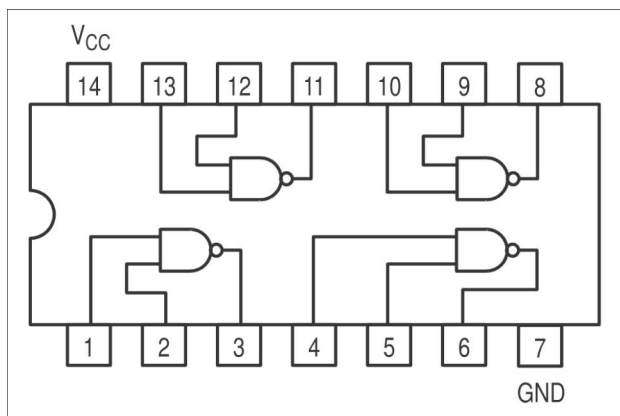



Figura 7

Gli integrati TTL

Sono stati i primi a essere realizzati negli Stati Uniti per motivi militari. La frequenza di lavoro massima è di 27 MHz e la tensione di alimentazione è fissa e del valore di 5V. La serie militare inizia con il numero 54 seguito altri numeri. Per esempio il 5400 corrisponde a un integrato che contiene quattro nand. Dato fondamentale, proprio perché utilizzati per motivi militari, la temperatura di utilizzazione va da -55° a 125°, mentre nella corrispondente serie civile, che inizia con i numeri 74, il range di temperatura va da -40° a +85° . Sono costruiti in formato DIP "Dual in-line package" (cioè i pin attraversano il circuito stampato o PCB). In figura 7 è mostrato l'integrato sopra citato. E' utile sottolineare che i suddetti circuiti integrati, realizzati a suo tempo principalmente dall'americana Texas Instruments e dalla giapponese National, sono ormai obsoleti. Li abbiamo illustrati brevemente sia per motivi storici, sia perché spesso incontriamo il termine "TTL compatibile", che vuol dire che il circuito che si sta trattando è

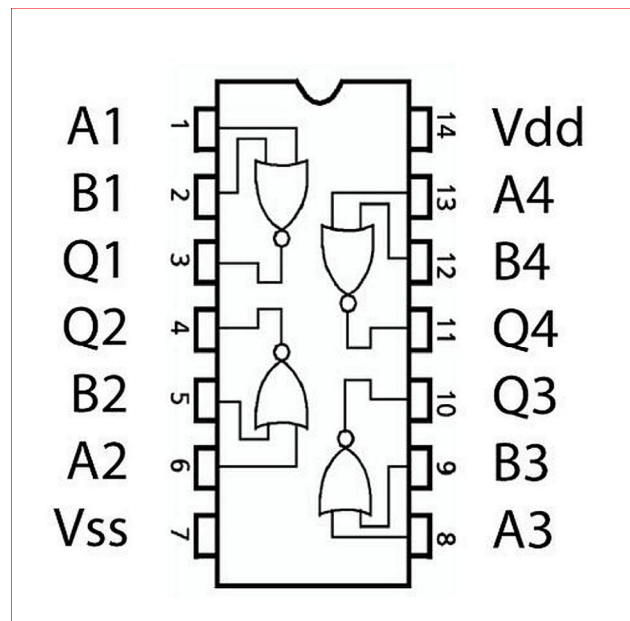



Figura 8

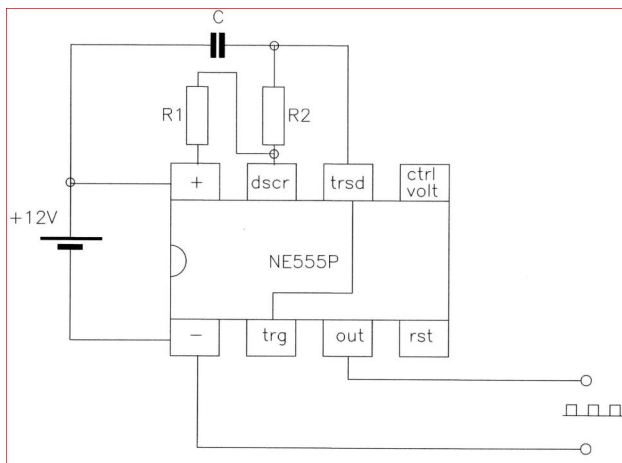



Figura 9

compatibile con altri che sono alimentati a +5V.

Gli integrati CMOS

Ancora attuale, anche se nata non molto dopo la 54/74 TTL, è la serie CMOS, acronimo di Complementary Metal-Oxide Semiconductor. Non è compito di questo articolo disquisire sulla tecnologia relativa alla realizzazione di tali componenti. Ci limitiamo a esaminare il loro comportamento elettrico (figura 8). A differenza dei TTL i CMOS si possono alimentare con una tensione che va da +3V a +18V ed hanno un bassissimo assorbimento di potenza. Il prezzo che pagano per tale caratteristica è la bassa velocità. Infatti, mentre i TTL lavoravano fino a 27 MHz, i CMOS non superano qualche MHz. Il basso consumo e la bassa velocità, che paradossalmente diviene una caratteristica positiva nelle applicazioni più comuni dato che per ciò risente meno dei disturbi della rete di alimentazione, unito al basso costo, insieme all'esteso range di tensione di alimentazione, lo rendono tuttora un tipo di componente molto utilizzato nell'elettronica, soprattutto industriale. L'NE555 come multivibratore astabile (generatore di onda quadra). Come esempio applicativo utilizzeremo un integrato molto noto agli appassionati di elettronica, l'NE555. Questo piccolo integrato, 8 pin DIP, ha enormi potenzialità ed un costo irrisorio. Iniziamo con il suo utilizzo come generatore di onda quadra analizzando il circuito di Figura 9. La formula che ci consente di determinare la frequenza del segnale d'uscita è la seguente:

$$f=1{,}44/(R1+2R2)C$$

Per avere una forma d'onda con l'intervallo di conduzione pari all'intervallo d'interdizione è necessario che R2 sia molto più grande di R1, per esempio 10 volte R1. Poniamo di voler ottenere in uscita un segnale f=100kHz con il valore di 12V. Fissiamo R1=1k, di conseguenza R2 dovrà essere 10k. Dalla formula sopra riportata ci ricaviamo il valore corrispondente di C:

$$C=1{,}44/(R1+2R2)f= \\ 1{,}44/1{,}1E3x1E6= \\ (1{,}44/1{,}1)uF= \\ 1{,}3uf\sim1{,}2uF$$

Se, com'è probabile, utilizziamo un condensatore elettrolitico, dobbiamo fare attenzione a connettere il polo positivo dello stesso al polo positivo dell'alimentazione. L'NE555 come multivibratore monostabile (generatore di un solo impulso). Il circuito che consente di realizzare quanto sopra esposto è mostrato in

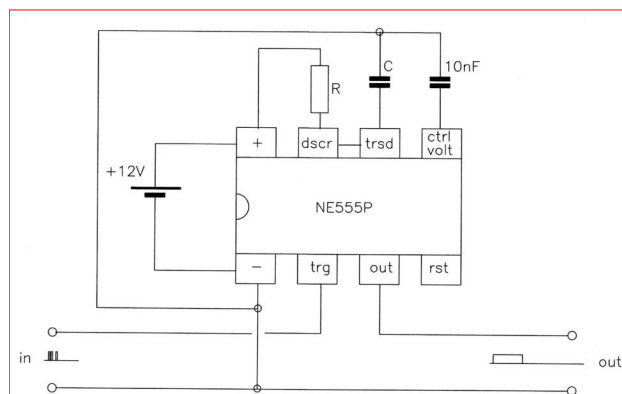



Figura 10

Figura 10. La formula che ci consente di determinare la durata dell'impulso di uscita è la seguente:

t=1,1RC

Il multivibratore monostabile è molto utile quando vogliamo applicare a un determinato circuito un solo impulso di una durata stabilita. L'esempio classico è quello di comandare un circuito elettronico tramite un comune pulsante.
Diversamente da come si può immaginare il pulsante premuto, prima di fermare la sua corsa verso il contatto finale genere una serie di impulsi.
Se noi applichiamo direttamente tale treno di impulsi al circuito da comandare si avrà una risposta indeterminata.
Poniamo che il transitorio, prima del contatto finale, sia di 300msec. Affinché il circuito da comandare riceva un solo impulso occorre che il monostabile generi un unico impulso della durata, per esempio, di 500msec. Dalla formula di cui sopra, utilizzando una resistenza da 100k ci ricaviamo il valore di C:

C=t/1,1 x R x t =
0,50/1,1 x E5=
0,454 x E-5=
4,54uF~4,7uF

Tante altre e importanti considerazioni ci sarebbero da fare su questo piccolo ma grande circuito integrato, senza ignorare che è uno tra i tantissimi CMOS esistenti e con le più svariate applicazioni.
Siamo partiti dal transistor e, strada facendo, siamo arrivati ai circuiti integrati che in un solo chip ne contengono migliaia.
Ci fermiamo per ora qui, sperando di avere fatto una panoramica sufficiente a dare l'idea delle potenzialità di questo piccolo, ma potentissimo, componente, dal quale non si può prescindere quando si tratta di elettronica, sia analogica che digitale.

Arduino da Zero: Telecomando a Infrarossi

di Davide Fiorino

Con questo articolo ci proponiamo di realizzare un progetto didattico leggermente più complesso rispetto a quello del primo episodio di "Arduino da Zero" (Fare Elettronica 354). Impareremo a usare Arduino non solo per ricevere informazioni dal mondo esterno, ma anche per inviarle e interagire con altri dispositivi elettronici, tramite la luce a infrarossi.

Descrizione del progetto

Quello che vogliamo realizzare è un telecomando a infrarossi con Arduino. Si tratterà di un telecomando universale poiché potremo utilizzarlo con qualsiasi dispositivo elettronico che usa la tecnologia di trasmissione a infrarossi (IR).
Nel nostro esempio lo useremo per controllare un decoder TV Samsung. Il progetto si articola in due fasi: la prima consiste nel decifrare e memorizzare i segnali IR di un telecomando e la seconda nel riprodurli per pilotare un determinato dispositivo.
Tramite il ricevitore IR potremo decifrare e analizzare praticamente qualsiasi tipo di segnale luminoso a questa frequenza (38 KHz).
Dal punto di vista software impareremo anche a importare una libreria esterna in Arduino e a crearne una nostra. Il progetto descritto in questo articolo è da intendersi come uno spunto per idee più complesse, oltre che a fini didattici.

Trasmissione IR

Il meccanismo di trasmissione a infrarossi si basa sulla modulazione di luce ad alta frequenza, circa 38 KHz per le applicazioni comuni. Il fascio luminoso inviato dal fotodiodo di un telecomando è opportunamente modulato in modo da creare un'alternanza di bit (stati 0-1). A ogni sequenza ordinata di bit corrisponde un comando preciso. Quando questo fascio luminoso arriva al dispositivo, il sensore fotoelettrico converte l'onda elettromagnetica in segnale digitale. Dal segnale digitale viene poi ricostruita la sequenza di bit di partenza e, se corrisponde a una di quelle conosciute dal dispositivo, viene eseguito il comando associato.
Il principio è abbastanza semplice e quando fu inventato permise di sostituire gli scomodi telecomandi a filo, negli anni '60. Nel nostro caso utilizzeremo Arduino, prima come ricevitore e poi come trasmettitore.

Componenti necessari

I componenti necessari alla realizzazione del nostro telecomando universale sono davvero pochi. Oltre a una scheda Arduino (nel nostro esempio Arduino Uno), utilizzeremo un ricevitore e un led a infrarossi per eseguire ricezione e trasmissione dei segnali. Ci sono diversi ricevitori IR sul mercato, noi utilizzeremo il modello IR38DM che è tarato sulla frequenza 38 KHz ed è dotato di preamplificatore, demodulatore e filtro. Il sensore è dotato di 3 pin: massa, tensione e segnale. Per la trasmissione invece ci servirà un

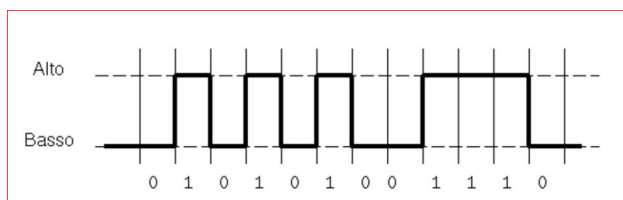



Figura 1: Esempio di segnale a sequenza binaria

semplicissimo led IR, come quelli presenti nei telecomandi. Avremo bisogno inoltre una resistenza da 100 Ω da mettere in serie al led. Sono necessari anche una breadboard e qualche cavetto per i collegamenti. Esclusa la scheda Arduino, i componenti hanno un costo di pochi euro.

Decodificare un segnale

La prima cosa che dovremmo fare per realizzare un telecomando è decodificare i segnali che usa il telecomando originale del dispositivo. Quindi se vogliamo pilotare un televisore, sarà necessario prendere il telecomando di questa TV e memorizzare, tramite il ricevitore IR collegato ad Arduino, tutti i codici comando che ci interessano.

Collegamenti elettrici

Prima di decodificare i segnali che ci interessano è necessario realizzare i collegamenti elettrici tra i componenti e la scheda Arduino. Come potete

Figura 2: Lo schema elettrico del progetto

vedere in figura 1, il ricevitore IR che ci interessa in questa fase del progetto, ha 3 pin. Il pin centrale (nel modello IR38DM) va collegato alla massa (GND) della scheda. Mettendo il sensore con la parte nera verso di noi, il pin di sinistra va connesso al pin 5V di Arduino e quello di destra al pin digitale 11. Se si utilizza un altro tipo di sensore si può consultare la scheda tecnica su Internet. Colleghiamo anche il LED IR: l'anodo va connesso al pin digitale 3 della scheda, facendolo passare prima dalla resistenza (100 Ohm), e il catodo a massa (GND). Il tutto dovrebbe risultare simile al rendering in figura 3.

Il programma per la ricezione

Passiamo adesso alla parte software: abbiamo bisogno di un programma in grado di mostrarci i segnali infrarossi ricevuti in un formato memorizzabile. Esiste già una libreria creata appositamente per questo tipo di progetti che si chiama IRremote ed è ovviamente open source. Quello che dovremmo fare è scaricarla dal sito ufficiale o da GitHub e installarla sulla IDE di Arduino. Prima di fare ciò però è necessario eliminare una libreria presente di default nella IDE, che va in conflitto con IRremote. La libreria in questione si chiama Robot IR Remote. Se usiamo un Mac, per disinstallarla bisogna aprire la cartella Applicazioni, fare tasto destro su Arduino e scegliere "Mostra contenuto pacchetto", aprire Contents > Java > libraries ed eliminare la cartella Robot_IR_Remote. Su Windows il procedimento è analogo: bisogna aprire la cartella libraries ed eliminare la libreria che va in conflitto. È sempre possibile riscaricare la suddetta libreria in un secondo momento nel caso possa servire. Per installare la libreria IRremote invece basta aprire l'IDE e andare sul menù Sketch > Include

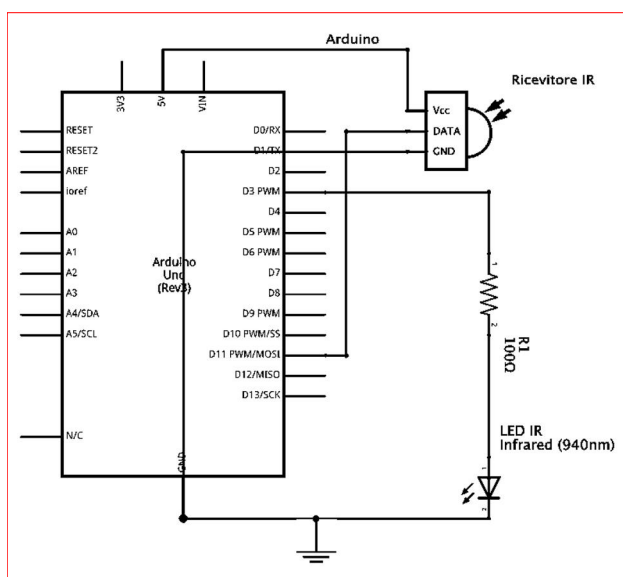
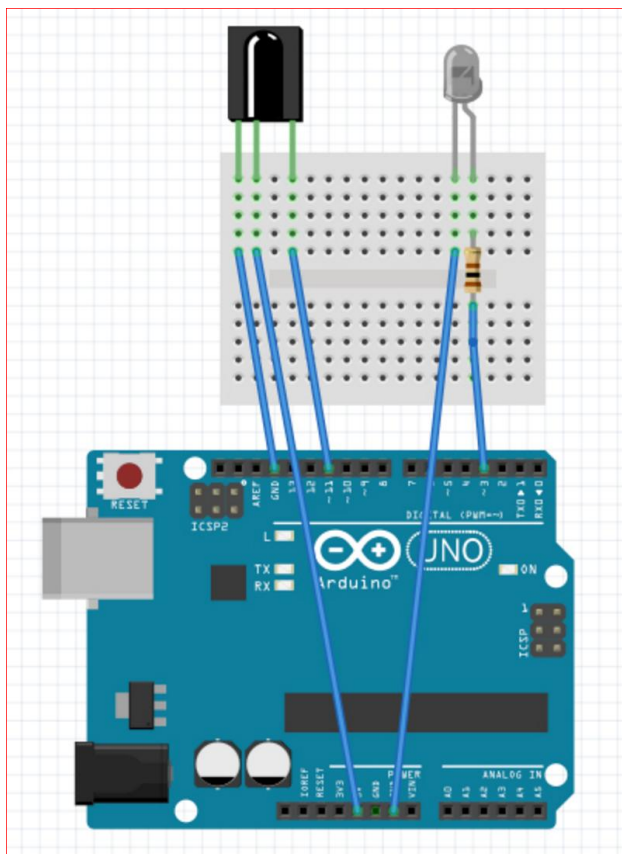



Figura 3: Il rendering del progetto con Fritzing

Library > Add .ZIP Library, e selezionare il file zip che avete scaricato da internet. Il programma (Listato 1) per ricevere e scrivere sul monitor seriale i codici di un telecomando, in realtà è già presente tra i file della libreria IRremote. Nella prima riga di codice si importa IRremote, e successivamente si imposta, tramite una funzione della libreria, il pin 11 (collegato al sensore IR) per la ricezione. Nelle righe successive troviamo una serie di condizioni "if" che fanno un controllo sul segnale ricevuto, per vedere se rispetta uno dei protocolli noti (NEC, Sony, Lg, JVC, ecc.). In caso contrario sul monitor seriale, il codice del comando sarà preceduto dalla scritta "Unknown encoding:" e verrà stampato per intero. Se si tratta di un protocollo conosciuto, il codice sarà, infatti, rappresentato in una forma abbreviata. Oltre al codice vero e proprio appariranno sul monitor la lunghezza del codice (in bit) e la lunghezza dell'array che lo contiene

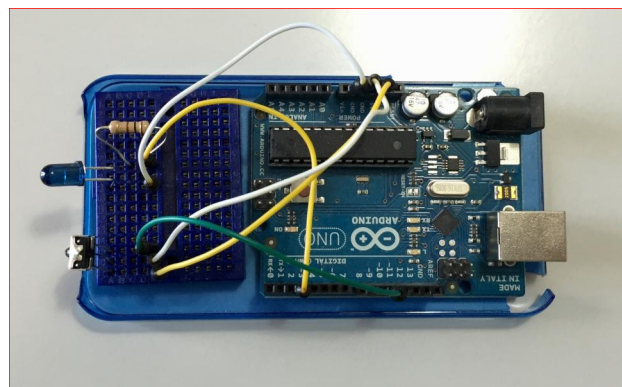



Figura 4: I vari componenti assemblati ad Arduino

nella sua forma intera, (solitamente intorno alle 70 posizioni). Una volta caricato il programma sulla scheda, puntando il telecomando verso il sensore IR e premendo un tasto, sul monitor seriale dovrebbe apparire qualcosa di simile alla figura 5. Dopo aver smanettato un po', copiamo i codici dei comandi che ci interessa riprodurre su un file (tipo blocco note), facendo attenzione a escludere il primo valore di ogni comando (Esempio: in figura 4, primo comando, "6666" sarebbe da escludere).

Inviare un segnale

Una volta salvati i codici comando che ci interessa riprodurre, passiamo alla parte successiva: la trasmissione. Per fare ciò è necessario creare un nuovo sketch di Arduino. Il programma in questione sarebbe però troppo lungo e confusionario se scritto tutto in un unico sketch, quindi decidiamo di suddividerlo e mettere tutti i codici comando in un file a se stante: una libreria.

Creare una libreria

Il mondo della programmazione di Arduino è strettamente legato alle librerie; tantissimi dispositivi compatibili con la scheda sono corredati da librerie software contenenti funzioni necessarie al loro funzionamento. La libreria è quindi un insieme di codice che fornisce

Figura 5: L'output del programma sul monitor seriale

determinate funzionalità. Nel nostro caso la funzionalità è quella di fornire i codici comando, sotto forma di array, quando richiesto. Nello sketch principale avremo quindi delle semplici chiamate a funzioni che ritornano questi codici. Una libreria su Arduino è composta da due file, uno con estensione .h (header file) e uno con estensione .cpp (source file). L'header file è, praticamente, una lista di tutte le funzioni, invece il source file contiene il codice vero e proprio. Nel nostro esempio abbiamo realizzato una libreria contenente i principali codici comando di un telecomando per decoder Samsung. Come si può vedere, nel listato 2 (header file) viene definita la classe con la parole chiave "class" e vengono dichiarati una serie di metodi pubblici, che corrispondono proprio ai comandi che ci interessa implementare. Nel source file invece (listato 3), bisogna importare l'header file e anche la libreria IRremote. Dopo aver creato un oggetto di tipo IRsend, mettiamo in alcuni array di numeri i codici comando che abbiamo precedentemente salvato. Per essere più precisi, nel listato 3 abbiamo implementato soltanto un comando (OnOff); chiaramente gli altri vanno implementati in modo analogo. Nell'ultima riga di codice definiamo l'apposita funzione per il comando OnOff, che non fa altro che chiamare il metodo sendRaw di IRremote e passargli tra i parametri l'array con il codice. Mettiamo adesso entrambi i file (.h e .cpp) in una cartella con nome analogo alla libreria (nel nostro esempio SamsugRemote) e posizioniamo la cartella all'interno di Documenti > Arduino > libraries.

Lo sketch finale

Una volta decodificati i codici del telecomando e memorizzati in un'apposita libreria, vediamo lo sketch finale, che "trasformerà" Arduino in un vero e proprio telecomando a infrarossi. Avendo suddiviso il programma in diversi file, questo sketch è davvero semplice. Come si può vedere dal listato 4 si inizia sempre importando le librerie che ci interessano e in questo caso anche quella contenente i codici comando, precedentemente scritta (nel nostro esempio SamsungRemote). Creiamo poi un'istanza di questa libreria, nel listato 4 chiamata "r". Questa istanza o oggetto, essendo di tipo SamsungRemote, avrà la capacità di eseguire tutte le funzioni da noi implementate nel source file della libreria. All'interno della funzione setup() avviamo semplicemente la comunicazione seriale a 9600 baud. I comandi, infatti, saranno impartiti scrivendo delle lettere nel monitor seriale. Nella funzione loop andiamo a memorizzare i caratteri inviati tramite il monitor seriale in una costante di tipo char (command). Analizziamo adesso il valore di questa costante tramite una struttura switch: se questa corrisponde a una delle lettere da noi scelte, inviamo il rispettivo comando. I comandi vengono inviati eseguendo le apposite funzioni scritte nella libreria SamsungRemote, che a loro volta non fanno altro che chiamare la funzione sendRaw di IRremote. Se tutto è stato fatto correttamente, il

nostro Arduino-telecomando è praticamente completo e funzionante. Basterà adesso collegarlo al PC tramite cavo USB, puntarlo verso il dispositivo che vogliamo pilotare e scrivere una lettera, fra quelle scelte, sul monitor seriale.

Creare un'interfaccia grafica

Il programma che abbiamo appena finito di scrivere è perfettamente funzionante, ma possiamo fare ancora qualcosa per migliorare il nostro telecomando. Si tratta di un'aggiunta facoltativa, anche perché al momento è compatibile soltanto con il sistema operativo Mac OS X. Non sarebbe molto più carino se invece di scrivere lettere sul monitor seriale si potesse pilotare Arduino cliccando su dei bottoni? Quello che faremo è quindi costruire una semplice interfaccia grafica (GUI), che renderà il tutto molto più intuitivo. Per fare ciò utilizzeremo Seriality, un plugin che consente al browser di inviare comandi al monitor seriale e quindi ad Arduino. Il plugin è stato sviluppato da Nicholas Zambetti, ed è compatibile con Safari, Chrome e Firefox su Mac. Scarichiamolo gratuitamente dall'indirizzo zambetti.com/projects/seriality, e installiamolo. Bisognerà adesso creare una pagina web (con estensione .html) che costituirà appunto la nostra GUI. Il codice (listato 5) è costituito da una parte in JavaScript che è proprio quello che ci permette di usare Seriality e da una parte in HTML con la quale è costruita la grafica. All'interno dello script troviamo la funzione serial.begin() con la quale vanno impostati la porta seriale (nell'esempio la 2) e la velocità in baud (9600). Nel caso non sappiate il numero della porta a cui avete collegato Arduino, basterà provare cambiando il numero in questione. Per creare i vari bottoni è sufficiente

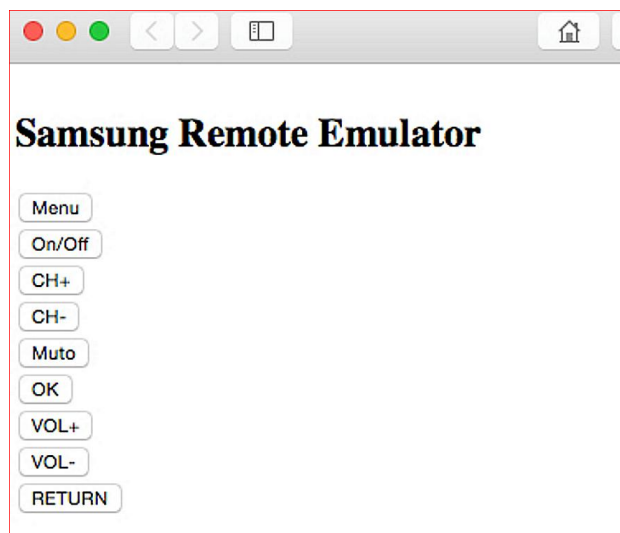



Figura 6: L'interfaccia grafica in HTML

copiare più volte il contenuto del tag <button> e modificare la scritta (ad esempio Menu) e la lettera corrispondente posta all'interno della funzione serial.write(). Salviamo il file html e apriamolo con un browser; il risultato dovrebbe essere simile a quello mostrato in figura 6. Cliccando sui bottoni, Arduino riceverà il comando e lo invierà al dispositivo tramite infrarossi.

Conclusioni

Questo progetto ci ha permesso di capire come, con Arduino, sia facile controllare anche dispositivi esterni di qualsiasi tipo. Le applicazioni derivanti da questo progetto possono essere svariate, anche nel campo della domotica. Si potrebbe ad esempio pensare di collegare la scheda ad Internet e usarla come telecomando di un condizionatore, per trovare la casa fresca quando arriviamo. Insomma, come al solito, con un po' di fantasia e dedizione, Arduino ci apre, in maniera semplice, le porte al mondo dell'elettronica.

Listato 1

```
#include <IRremote.h>
int RECV_PIN = 11;

IRrecv irrecv(RECV_PIN);
decode_results results;

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
}

void dump(decode_results *results) {
  int count = results->rawlen;
  if (results->decode_type == UNKNOWN) {
    Serial.print("Unknown encoding: ");
  }
  else if (results->decode_type == NEC) {
    Serial.print("Decoded NEC: ");
  }
  else if (results->decode_type == SONY) {
    Serial.print("Decoded SONY: ");
  }

  else if (results->decode_type == LG) {
     Serial.print("Decoded LG: ");
  }
  else if (results->decode_type == JVC) {
     Serial.print("Decoded JVC: ");
  }
  else if (results->decode_type == SAMSUNG) {
     Serial.print("Decoded SAMSUNG: ");
  }

  Serial.print(results->value, HEX);
  Serial.print(" (");
  Serial.print(results->bits, DEC);
  Serial.println(" bits)");
  Serial.print("Raw (");
  Serial.print(count, DEC);
  Serial.print("): ");

for (int i = 0; i < count; i++) {
    if ((i % 2) == 1) {
     Serial.print(results->rawbuf[i]*USECPERTICK, DEC);
    }
    else {
     Serial.print((int)results->rawbuf[i]*USECPERTICK, DEC);
    }
    Serial.print(",");
  }
 Serial.println("");
}

void loop() {
  if (irrecv.decode(&results)) {
   Serial.println(results.value, HEX);
    dump(&results);
    irrecv.resume(); // Receive the next value
  }
}
```

Listato 2

```
//SamsungRemote.h
#ifndef SamsungRemote_h
#define SamsungRemote_h

#include "Arduino.h"

class SamsungRemote
{
  public:
    SamsungRemote();
    void onOff();
    void muto();
    void chMin();
    void chPlus();
           void volMin();
           void volPlus();
           void ok();
           void ret();
           void menu();
  private:

};
#endif
```

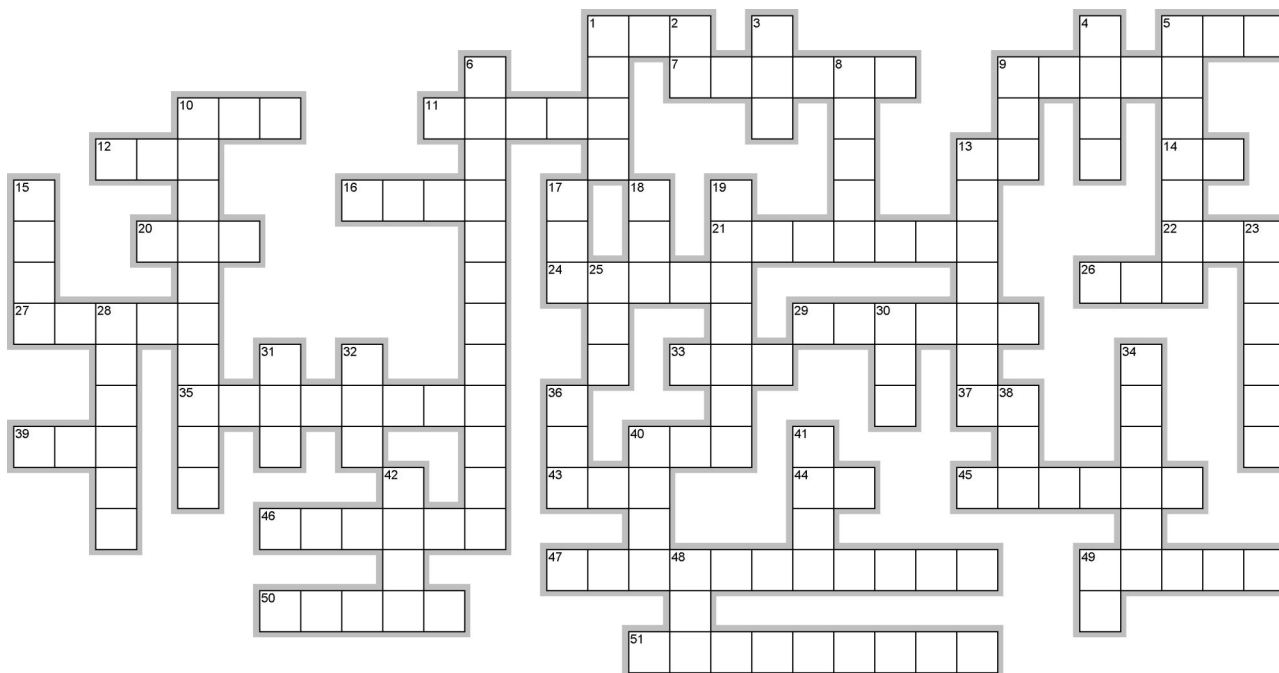
Listato 3

```
//SamsungRemote.cpp
#include "Arduino.h"
#include "SamsungRemote.h"
#include <IRremote.h>

SamsungRemote::SamsungRemote()
{

}
IRsend sender;

unsigned int OnOff[78]= {4400,4450,450,500,450,550,500,ecc…};

void SamsungRemote::onOff() {sender.sendRaw(OnOff,78,38);}
```

Listato 4

```
#include <SamsungRemote.h>
#include <IRremote.h>
#include <IRremoteInt.h>

SamsungRemote r;

void setup()
{
  Serial.begin(9600);

}

void loop() {
  if(Serial.available()) {
    const char command = Serial.read();

switch(command) {
     case 'o':
       r.onOff();
       break;
     case'm':
       r.menu();
       break;
    }
  }
}
```

Listato 5

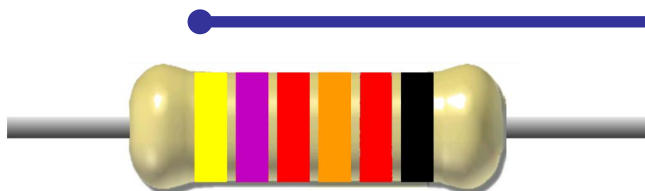
```
<html>
  <title>Samsung Remote Emulator</title>
  <head>
    <script type="text/javascript">
      var serial;
      function setup() {
        serial=(document.getElementById("seriality")).Seriality();
        alert(serial.ports.join("\n"));
        serial.begin(serial.ports[2], 9600); //selezionare la porta seriale(2)
      }
    </script>
  </head>
  <body onload="setup();">
    <object type="application/Seriality"
            id="seriality"
            width="0"
            height="0">
    </object>
    <h2>Samsung Remote Emulator</h2>
<form>
      <button type="button" onclick="serial.write('m');">
        Menu
      </button>
      <br/>
      <button type="button" onclick="serial.write('o');">
        On/Off
      </button>
      <br/>
</form>
  </body>
</html>
```

CROSSWORD

EclipseCrossword.com

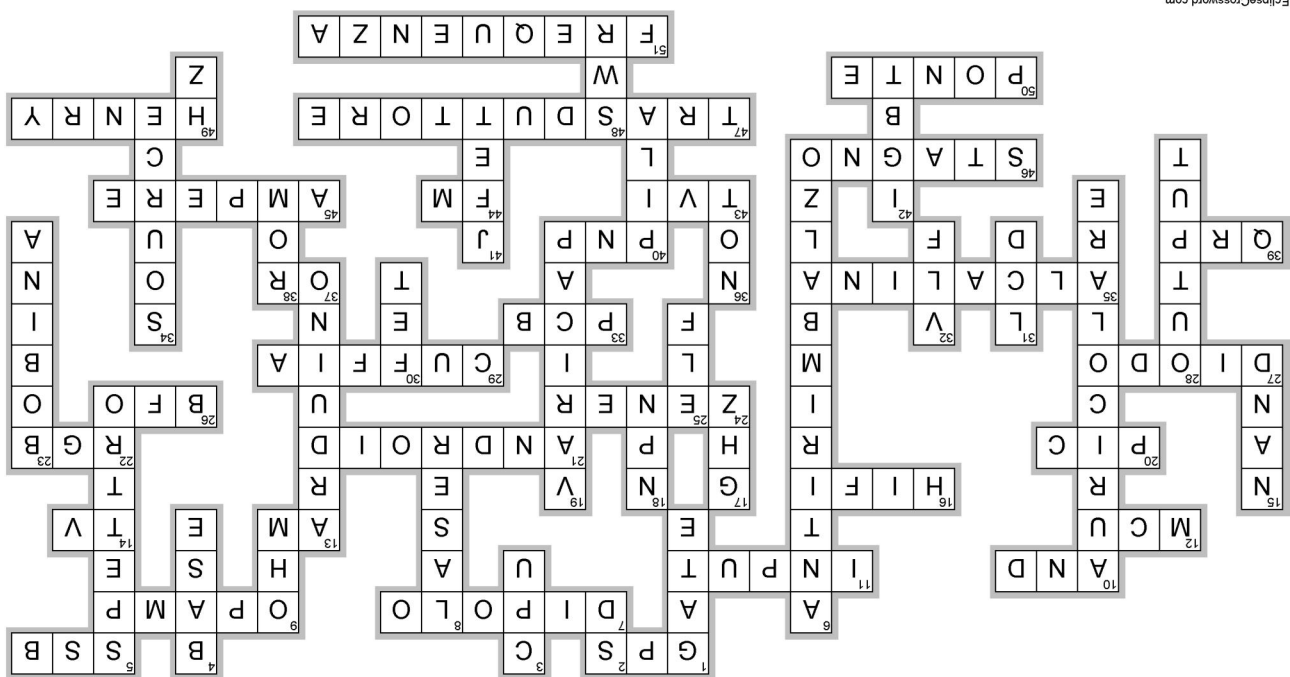
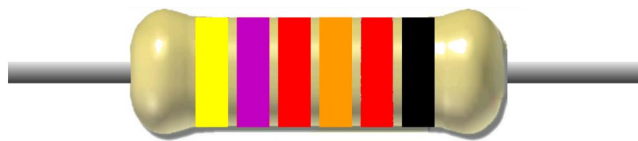
ORIZZONTALI

1. Sistema di Posizionamento Globale
5. Banda laterale singola
7. Antenna costituita da due bracci uguali
9. Abbreviazione di operazionale
10. Operatore Booleano
11. Ingresso di un circuito
12. Microcontrollore
13. Modulazione di ampiezza
14. Televisione
16. Alta fedeltà
20. Faniglia di microcontrollori
21. Sistema operativo per dispositivi mobili
22. Rosso, verde e blu
24. Tipologia di diodi
26. Oscillatore a frequenza di battimento
27. Dispositivo elettronico a due elettrodi che consente il passaggio di corrente in una sola direzione
29. Accessorio per l'ascolto individuale di suoni
33. Circuito stampato
35. Tipologia di pila
37. Operatore Booleano
39. Emissione a bassa potenza
40. Tipologia di transistor
43. Interferenze televisive a causa di trasmettitori radio vicini
44. Modulazione di frequenza
45. Unità fondamentale di misura dell'intensità di corrente elettrica
46. Serve per la brasatura
47. Converte l'energia da una forma ad un altra
49. Unità di misura dell'induttanza
50. Struttura circuitale a quattro lati
51. Numero di cicli che una grandezza variabile compie ll'unità di tempo

VERTICALE

1. Un terminale del transistor FET
2. Memoria per fotocamera e altro
3. Unità di processo centrale
4. Un terminale del transistor
5. Banda continua di radiazioni elettromagnetiche
6. Produce un solo impulso stabile in presenza di un ingresso elettricamente rumoroso
8. Consente di ottenere fasci intensi di luce
9. Unità di misura della resistenza elettrica
10. Dispositivo per ascolare suoni e musica in un orecchio
13. Schedina elettronica con un microcontrollore e circuiteria di contorno, utile per creare rapidamente prototipi e per scopi hobbistici e didattici
15. Operatore Booleano
17. Un miliardo di Hz
18. Tipologia di transistor
19. Diodo capace di variare la propria capacità interna
23. Filo avvolto a spire su un supporto cilindrico
25. Extremely low frequency
28. Uscita del segnale
30. Transistore a effetto campo
31. Display a cristalli liquidi
32. Generatore di energia elettrica
34. Un terminale del transistor FET
36. Operatore Booleano
38. Memoria non volatile
40. Generatore di energia elettrica
41. Transistor ad effetto di campo a giunzione
42. Transistor bipolare a gate isolato
48. Onde stazionarie
49. Unità di misura della frequenza

Di che resistore si tratta ?

Soluzioni

CROSSWORD

RICONOSCI IL RESISTORE

Si tratta di un resistore a 6 anelli di:
472 kΩ, tolleranza +/- 2%, Coefficente di temp: 200 ppm/K
Valore minimo: 462,56 kΩ - Valore massimo: 481,44 kΩ

- Banda 1: giallo	4
- Banda 2: viola	7
- Banda 3: rosso	2
- Banda 4: arancione	x 1000
- Banda 5: rosso	2%
- Banda 6: nero	200 ppm/K

Banana Pi diventa un potente e sicuro Cloud Server

di Ivan Scordato

Grazie alla tecnologia che ogni giorno compie passi da gigante, viviamo in un mondo collegato in tutti i campi: dai Social Network, che mettono in contatto persone conosciute e non, ai siti web di E-commerce che permettono a chiunque di trovare il miglior prodotto al prezzo minore, ai sistemi di sorveglianza ormai collegati in rete, fino ad arrivare ai Cloud Storage. Sia gli utenti privati che le aziende, specialmente negli ultimi tempi, hanno cominciato a sentire la necessità di poter accedere da qualunque parte del mondo ai propri dati, senza dovere portarsi dietro un hard disk o supporti di memoria. La soluzione a questa esigenza sta proprio nel Cloud Storage, che a volte però non offre esattamente quello di cui si ha bisogno, ed è per questo che la migliore soluzione è quella di realizzare un sistema che abbia le caratteristiche di cui si ha bisogno. Vediamo insieme come realizzarne uno con il Banana Pi.

Come abbiamo visto nell'articolo precedente, il Banana PI è una piattaforma hardware open source creata dal Team cinese LeMaker con lo scopo di promuovere la sviluppo e la ricerca in diversi ambiti, attirando comunque l'attenzione di molti curiosi e Maker, visto che permette di realizzare progetti fantastici. I motivi che hanno fatto nascere la necessità di avere i propri dati su un server collegato in rete sono tanti, tra cui il più importante è quello di avere la sicurezza che i propri dati non si smarriscano.
Il Cloud Storage è un tipo di conservazione dei dati che avviene tramite l'impiego di un computer collegato in rete che permette di incrementare la sicurezza dei propri file e l'affidabilità, tutto in un solo servizio che permette l'accesso ai file stessi solo a coloro che hanno le credenziali per accedervi.
Esistono molti servizi, sia a pagamento che gratuiti che offrono questa possibilità, ma spesso non offrono esattamente quello di cui si ha bisogno e comunque è meglio avere un Server personale sul quale salvare i propri dati in modo da essere sicuri sulla velocità, gestione e inattacabilità del sistema. La soluzione che ho scelto per realizzare un Cloud Storage è quella di utilizzare un Banana Pi tramite il quale è possibile ottenere un sistema molto compatto e ad un costo molto basso.
Caratteristica che rende il Banana Pi ancora più invitante per realizzare questo progetto è la presenza di un connettore SATA sulla board tramite il quale è possibile collegare direttamente un hard-disk alimentatandolo tramite il connettore POWER sul quale memorizzare i file, senza avere bisogno di utilizzare convertitori SATA-USB.

Realizzazione

Il Banana Pi che andremo ad utilizzare per realizzare il nostro Cloud Server personale necessità di una configurazione Hardware e Software abbastanza semplice da realizzare.

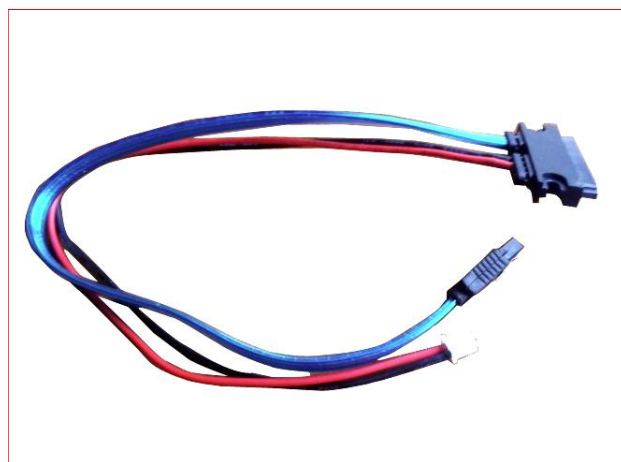



Figura 1: Connettore SATA e POWER sul Banana

Per realizzare questo progetto basta infatti avere poco materiale e un po' di pazienza per l'assemblaggio e il collegamento di tutto il necessario. Adesso vediamo come procedere per quello che riguarda la parte Hardware e Software.

Parte Hardware

L'hardware di cui abbiamo bisogno per realizzare il nostro Cloud Storage è il seguente:

- Banana Pi
- Hard Disk USB o SATA se si vuole utilizzare il connettore sul Banana Pi
- Connettore SATA e cavetto di alimentazione bipolare se si vuole utilizzare il connettore sul Banana Pi
- Cavetto LAN o Dongle WIFI
- Alimentatore stabilizzato 5V
- Pulsante switch
- 1 LED rosso
- 1 LED verde
- 1 LED blu
- Scatola dove alloggiare il Banana Pi e i componenti

Per fare in modo che il tutto risulti essere compatto e con una buona estetica, basterà utilizzare un contenitore nel quale alloggiare il Banana Pi e tutto quello di cui abbiamo bisogno.
Personalmente ho preferito recuperare il contenitore esterno di un vecchio ricevitore satellitare, ottenendo un risultato eccezionale sia in termini di estetica che di funzionalità. Utilizzeremo due LED per indicare lo stato del sistema e un pulsante per accendere e spegnere il server LAMP, in modo da poterlo disattivare quando non ci serve. Per collegare il pulsante e i LED al Banana Pi è necessario utilizzare i pin GPIO presenti sulla scheda stessa, tramite i quali è possibile collegare qualsiasi tipo di sensore e modulo che

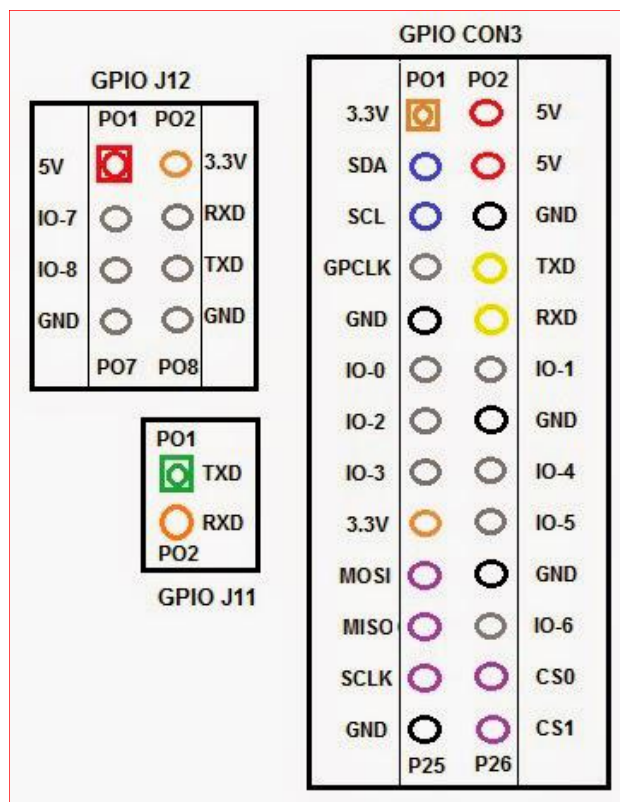



Figura 2: Pin GPIO del Banana Pi

ci serve per realizzare i nostri progetti; ad esempio possiamo collegare un sensore di temperatura, una scheda di espansione UART, etc...
Il PinOut del connettore GPIO del Banana Pi è mostrato in figura 2. Per collegare il pulsante e i LED al Banana Pi bisogna seguire lo schema di figura 3.
I file del server verranno salvati su un Hard Disk esterno.
Oltre ad utilizzare un Hard Disk con interfaccia USB è possibile utilizzarne direttamente uno sfruttando il

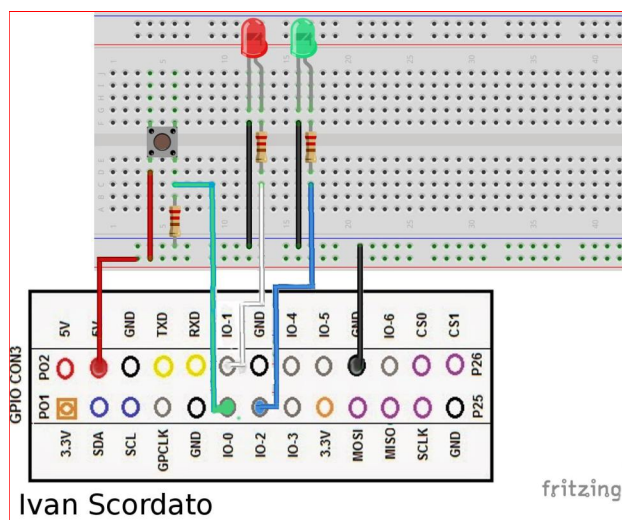



Figura 3: Collegamento Pulsante e Led

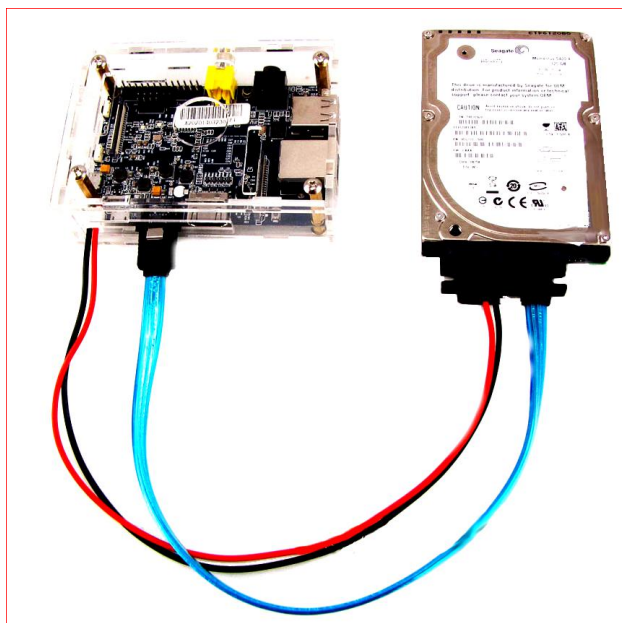



Figura 4: Collegamento Hard Disk

connettore SATA, collegandolo direttamente al Banana Pi tramite un cavetto apposito, come nell'immagine seguente. Nonostante questa accattivante caratteristica del Banana Pi, per il momento sono costretto ad utilizzare un Hard Disk collegato alla porta USB. Dopo aver eseguito questi collegamenti ed avere collegato al Banana Pi l'alimentatore e tutti gli altri accessori necessari al suo funzionamento, non resta che sbizzarrirsi sul come assemblare il tutto nel box che si è scelto di utilizzare.

Parte Software

Le operazioni da portare a termine per realizzare il sistema sono le seguenti:

•Impostare indirizzo IP del Banana Pi in modalità statico;
•Installare la piattaforma LAMP tramite la quale è possibile sviluppare applicazioni Web;
•Installare un Server FTP;
•Caricare sul server locale il sistema di Cloud Storare e la piattaforma phpMyAdmin;
•Configurazione OwnCloud;
•Settaggio protocollo HTTPS e ultime configurazioni di Apache;
•Installare le librerie per la gestione dei pin GPIO e programmazione in Python per l'accensione/spegnimento dei led e lettura stato del pulsante;
•Registrazione di un dominio DNS.

Impostare l'indirizzo IP

La maggior parte delle volte il router assegna ad ogni dispositivo collegato in rete, un indirizzo IP casuale che cambia ogni volta che il dispositivo effettua un nuovo collegamento al router stesso. Per il progetto che stiamo realizzando questo può essere un problema, e quindi è necessario cambiare l'impostazione dell'indirizzo IP del nostro Banana Pi da dinamico a statico, in modo da sapere sempre a quale indirizzo IP raggiungerlo. Per effettuare questa operazione dobbiamo controllare qual'è il BroadCast, l'Host e La Mask della rete, dati ottenibili digitando semplicemente "ifconfig" sul Terminale. Dopo aver appuntato questi indirizzi ed avere eseguito l'accesso come utente root (digitando "sudo –i"), basta modificare il file di configurazione dell' interfaccia network nel seguente modo:

auto lo

iface lo inet loopback

iface eth0 inet static

address 192.168.1.70 ==> L'indirizzo IP che vogliamo assegnare al nostro Raspberry Pi
netmask 255.255.255.0 ==> Quì la Netmask

gateway 192.168.1.254 ==> Quì inseriamo il Gateway del Router

network 192.168.1.0 ==> Quì scriviamo l'indirizzo del Network

broadcast 192.168.1.255 ==> Quì inseriamo il Broadcast

Figura 5: Sistema assemblato

allow-hotplug wlan0

iface wlan0 inet manual

wpa-roam
/etc/wpa_supplicant/wpa_supplicant.conf

iface default inet dhcp

Digitiamo sul terminale:

```
nano /etc/network/interfaces
```

e modifichiamo il file.

Sempre dopo avere eseguito l'accesso come utente root, per installare il server LAMP basterà digitare direttamente nel terminale il seguente comando:

```
apt-get install apache2 php5
libapache2-mod-php5
```

Riavviamo Apache:

```
sudo service apache2 restart
```

Installazione LAMP
LAMP è un acronimo che prende il nome dalle iniziali dei componenti software con cui è realizzata, cioè: Linux - Apache - MySQL – PHP. Per attivare HTACCESS è indispensabile modificare il file config digitando il comando nano /etc/apache2/sites-enabled/000-default, e sostituiamo "AllowOverride None" con "AllowOverride ALL".
Per migliorare l'accessibilità al sistema, e quindi per poter eseguire l'accesso al nostro server tramite il protocollo FTP è necessario installare un FTP Server.E' possibile trovarne diversi che si differenziano per alcune piccolezze. Quello che mi sento di consigliare è il VSFTP. Prima di procedere all'installazione, è necessario impostare il percorso nel quale abbiamo deciso di inserire la cartella www di Apache, che nel nostro caso si trova nel nostro hard disk esterno. E' altresì indispensabile impostare i relativi permessi della cartella. Prima di tutto spegniamo il servizio Apache:

```
sudo service apache2 stop
```

Adesso possiamo modificare il file di configurazione di Apache, aprendolo direttamente con gedit per la modifica, quindi digitiamo sul terminale:

```
sudo gedit
/etc/apache2/sites-
available/default
```

A questo punto cercate queste due righe nel file:

DocumentRoot /var/www/
Directory /var/www/

e sostituitele con l'indirizzo della cartella WWW che avete creato nel vostro Hard Disk.

Ora basta riavviare il servizio di Apache digitando nel terminale:

```
sudo service apache2 start
```

Una volta riavviato, Apache considererà la nuova cartella come DocumentRoot, e da adesso in poi dovremo inserire tutti i file in questa

cartella. Per vedere se tutto funziona a dovere, basta aprire il browser direttamente dal Banana Pi e digitare, come indirizzo sulla barra di navigazione, l'indirizzo IP che abbiamo assegnato al nostro sistema o più semplicemente “localhost”.

Installazione phpMyAdmin

Adesso che il nostro Server locale è stato settato correttamente, possiamo procedere con la configurazione della parte WEB del nostro Cloud Storage. Per creare un sistema funzionale, completo e sicuro, non possiamo fare a meno di utilizzare MySQL, tramite il quale utilizzando una serie di database e tabelle possiamo catalogare e gestire i file che andremo a caricare sul Cloud e le credenziali d'accesso per gli utenti autorizzati ad accedere al sistema. Abbiamo già installato il servizio MySQL in precedenza quando ci siamo occupati del server LAMP, ma per potere avere un sistema più dinamico, è necessario utilizzare un'applicazione Web che ci permetta di gestire con facilità I nostri database, come ad esempio il famoso ed eccezionale phpMyAdmin. PhpMyAdmin è un'applicazione web rilasciata con licenza GPL, scritta in PHP, tramite la quale è possibile amministrare un database MySQL utilizzando qualunque browser. PhpMyAdmin mette a disposizione dell'utilizzatore molte funzioni, tra le quali quella di creare un database e tabelle da zero e ottimizzare le stesse. Non mancano nemmeno degli strumenti indispensabili e più particolari che permettono di svolgere con estrema facilità le operazioni più delicate, come ad esempio la popolazione del database, il backup dei dati, ecc. Quello che adesso andremo a fare non può essere considerata come un'installazione vera e propria, in quanto per utilizzarlo, trattandosi di un insieme di pagine PHP, basta semplicemente scaricare phpMyAdmin dal suo sito ufficiale e decomprimere l'archivio in una cartella del proprio server web (nel nostro caso nella cartella www). Questa operazione può essere semplificata tramite il sistema di gestione dei pacchetti presente nella maggior parte delle distribuzioni Linux, che nel nostro caso ci permette di installare facilmente phpMyAdmin in una volta sola, digitando sul terminale, dopo aver effettuato l'accesso come utente root, il seguente comando:

```
apt-get install phpmyadmin
```

Dopo avere confermato lo scaricamento dei pacchetti, bastano pochi minuti per il completamento dell'operazione, dopo la quale possiamo aprire il browser all'indirizzo localhost/phpMyAdmin per configurare l'applicazione.

Installazione e configurazione OwnCloud

Adesso possiamo proseguire con la creazione Web del Cloud Storage. Essenzialmente quello di cui abbiamo bisogno è un sistema che ci permetta di potere caricare dei file, organizzarli in cartelle e settare i permessi per gli utenti che inseriremo. Ci si trova a dover fare una scelta: creare da se il proprio sistema, o sceglierne uno già creato da qualcuno. La scelta dell'alternativa è vincolata dalle proprie capacità di programmazione o

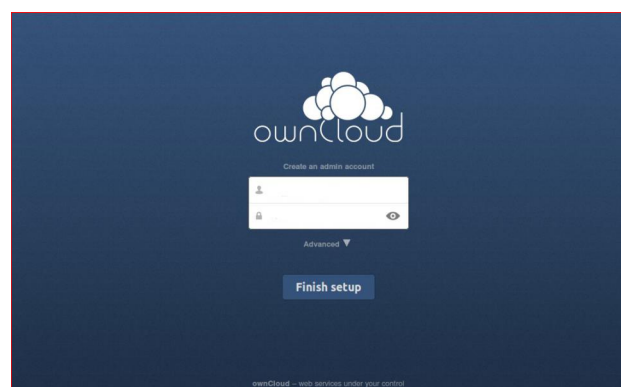



Figura 6: Schermata configurazione OwnCloud

Figura 7: Seconda schermata di configurazione OwnCloud

più semplicemente dal fatto che si preferisca utilizzare un sistema già pronto che è comunque possibile modificare. Personalmente ho sperimentato entrambe le soluzioni, ottenendo in entrambi i casi un ottimo risultato; attualmente il mio lavoro non lo renderò pubblico in quanto sono ancora presenti dei bug che non renderebbero il sistema al 100% sicuro, ma consiglio di utilizzare ownCloud, nato come un'applicazione di file hosting che non permette soltanto di caricare file e documenti, ma anche di crearne di nuovi tramite un editor online e di condividerli in tempo reale, e include anche una serie di plugin con i quali è possibile gestire i propri contatti, il calendario e molte altre attività professionali. Dopo avere scaricato l'ultima versione di OwnCloud dal sito ufficiale (owncloud.org), bisognerà scompattare direttamente l'archivio nella cartella www presente nell'Hard Disk che stiamo utilizzando per il progetto. La prima cosa da fare è quella di configurare OwnCloud, operazione semplice che si effettua aprendo il browser e andando su localhost, dove ci si troverà davanti a una schermata simile alla seguente. In questa pagina ci viene chiesto di scegliere un nome utente ed una password d'accesso, che naturalmente sono indispensabili per accedere a OwnCloud. Con le impostazioni di default, per la gestione del database, ownCloud utilizzerà SQLite, ma noi che in precedenza abbiamo installato MySql, possiamo scegliere di cambiare il database da utilizzare, in modo da avere, in teoria, un sistema più affidabile su volumi di dati maggiori. Dopo avere terminato la configurazione, basterà cliccare in basso sul pulsante Termina la Configurazione, e la prima cosa che vedremo sarà la finestra di benvenuto che ci indicherà la possibilità di collegare la piattaforma Web ad altri sistemi. Da questo momento in poi potremo utilizzare l'interfaccia di OwnCloud, che risulta essere molto intuitiva e molto efficace, infatti sul lato sinistro è presente la colonna con la suddivisione dei file in base all'appartenenza, mentre nello spazio a destra è possibile visualizzare i file presenti nel nostro spazio d'archiviazione. E' presente inoltre un menu che permette di accedere ai servizi ownClod e una comoda barra di ricerca, utile per trovare quello che cerchiamo sia in locale che sugli altri device connessi. Un aspetto molto importante che non bisogna sottovalutare, è la cifratura per tutte le connessioni del server e dei client; per farlo è necessario configurare openSSL e consentire l'uso del protocollo HTTPS, procurandoci un certificato "self-signed".

Digitiamo sul terminale:

```
sudo apt-get install openssl
sudo a2enmod ssl
sudo mkdir /etc/apache2/ssl
```

Dopodichè digitiamo:

```
sudo openssl req -x509 -nodes
-days 365 -newkey rsa:2048 -keyout
/etc/apache2/ssl/owncloud.key -out
/etc/apache2/ssl/owncloud.crt
```

In questo modo il sistema di generazione della chiave di

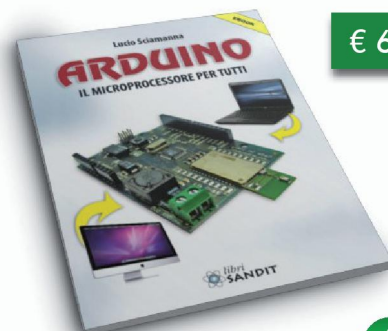
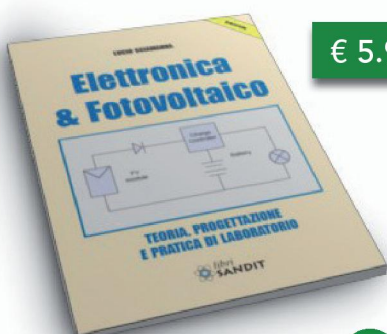
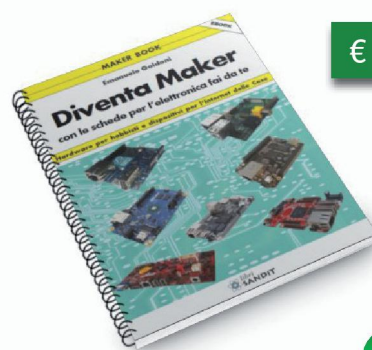
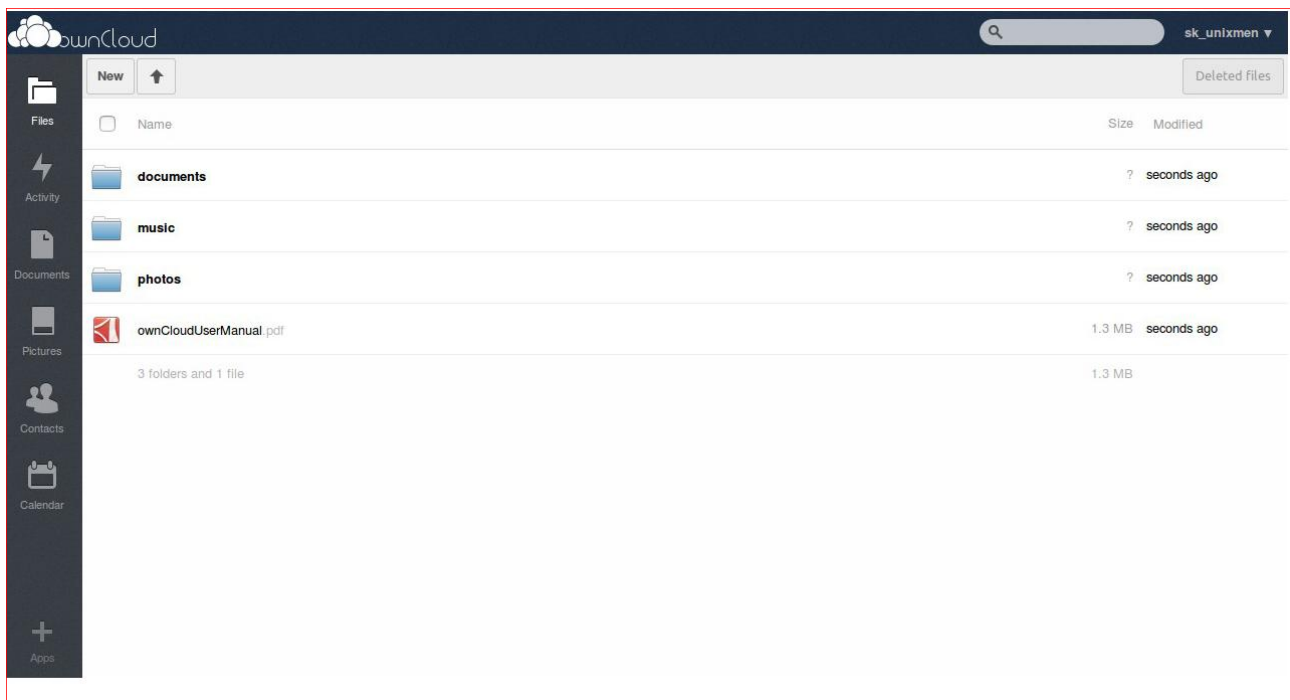



Figura 8: OwnCloud Pronto per essere utilizzato

crittografia e dei certificati ci chiederà alcune informazioni che possiamo tranquillamente inventare, tranne il CommonNow. Per sintetizzare e semplificare, ecco cosa possiamo inserire:

- In Country Name scriviamo IT;
- In State scriviamo Italia;
- Locality Name possiamo lasciarlo vuoto;
- Organization Name inventiamoci un nome;
- Organization Unit Name possiamo lasciarlo vuoto;
- Nel campo Common Name inseriamo il nostro IP pubblico, cioè il nostro indirizzo statico che abbiamo in precedenza impostato;
- Inseriamo il nostro indirizzo email.

Dopo questa operazione non bisogna fare altro che abilitare il protocollo SSL con il certificato appena creato modificando alcune righe del file di configurazione, perciò digitiamo sempre sul terminale:

```
sudo gedit /etc/apache2/sites-available/default-ssl.conf
```

Nell'editor di testo troviamo le corrispondenti righe da modificare:

ServerName IPstaticoimpostato
SSLEngine on
SSLCertificateFile
/etc/apache2/ssl/owncloud.crt
SSLCertificateKeyFile
/etc/apache2/ssl/owncloud.key

Se dovesse mancare qualche riga (ad esempio ServerName) basterà aggiungengerla nel primo spazio libero, mentre se in alcune righe i parametri già presenti non dovessero corrispondere a quelli che ho indicato, dovrete correggerli (probabilmente SSLCertificateFile e SSLCertificateKeyFile).

Salviamo il file e continuiamo la configurazione attivando il protocollo SSL su Apache, digitando sempre sul terminale:
sudo a2ensite default-ssl
Riavviamo Apache per rendere effettive le modifiche:
sudo service apache2 restart

Se tutto è andato a buon fine, da questo momento in poi potremo

```
test@test-VirtualBox:~$ sudo openssl req -newkey rsa:2048 -x509 -days 3650 -node
s -out /etc/ssl/certs/apache.pem -keyout /etc/ssl/certs/apache.key
Generating a 2048 bit RSA private key
.........................................................................
.............................+++
..........................................................................+++
writing new private key to '/etc/ssl/certs/apache.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:IT
State or Province Name (full name) [Some-State]:Italia
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Private
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:Unico
Email Address []:
test@test-VirtualBox:~$
```

Figura 9: Configurazione parametri

accedere all'interfaccia di configurazione di ownCloud usando il protocollo HTTPS. Aprendo il browser, dopo esserci recati all'indirizzo https://localhost, verremo informati che il certificato non è valido, perciò aggiungiamo tranquillamente un'eccezione permanente per ownCloud.
Di default ownCloud consente l'accesso solo dal server stesso (localhost) e non permette ad altri PC di connettersi al server. Per eliminare questo limite e permettere quindi agli altri PC della stessa rete di connettersi e per garantire l'accesso tramite Internet da qualunque parte del mondo, dobbiamo modificare il file PHP "config.php", digitando sul terminale:

```
sudo gedit
/var/www/owncloud/config/config
.php
```

Individuiamo la sezione trusted_domains:

```
'trusted_domains' =>
 array (
 0 => 'localhost',
 ),
```

Modifichiamola come segue:

```
'trusted_domains' =>
 array (
 0 => 'localhost', 1 => 'IndirizzoIP_LAN', 2 => 'IndirizzoIP_Publico',
 ),
```

Al posto di IndirizzoIP_LAN inseriamo l'indirizzo IP del Banana Pi connesso alla rete locale, in modo da garantire l'accesso agli altri device della stessa rete. Al posto di IndirizzoIP_Publico inseriamo l'indirizzo IP pubblico del nostro Banana Pi, così da garantire l'accesso ad altri device connessi tramite Internet.

Accesso tramite il World Wide Web

Per accedere al sistema da qualunque parte del mondo tramite un browser, è necessario registrare un dominio DNS ed impostare tramite la pagina di configurazione del router l'indirizzo dns che abbiamo creato. E' possibile registrare un dominio dns tramite molti

portali, tra i quali, mi sono trovato particolarmente bene con dyndns.it.

Utilizzo GPIO

Per utilizzare i pin GPIO del Banana Pi e potere rilevare la pressione del pulsante e accendere e spegnere i led, è necessario utilizzare la libreria RPi.GPIO, creata originariamente per il Raspberry Pi e modificata in seguito. Può essere scaricata direttamente dallo spazio Github di LeMaker. Per chi possiede il Banana Pi, bisogna digitare sul terminale:

```
git clone
https://github.com/LeMaker/RP
i.GPIO_BP -b bananapi
```

Mentre chi utilizza un Banana Pro, deve digitare sul terminale:

```
git clone
https://github.com/LeMaker/RP
i.GPIO_BP -b bananapro
```

Per installare la libreria è necessario avere il pacchetto python-dev, che è possibile ottenere digitando sul terminale:

```
sudo apt-get update
sudo apt-get install python-dev
```

E in seguiro compilare ed installare il tutto, digitando sul terminale quanto segue:

```
cd RPi.GPIO_BP
python setup.py install
sudo python setup.py install
```

Adesso è possibile utilizzare lo script python RPi.GPIO.

WiringPi

Wiring Pi è una libreria GPIO scritta da Drogon, creata originariamente per il Raspberry Pi, ma il team LeMaker l'ha modificata e adattata per farla funzionare sul Banana Pi e Pro, e

```
Fichier Édition Affichage Rechercher Terminal Aide
pi@bananapi ~ $ gpio -v
gpio version: 2.14
Copyright (c) 2012-2014 Gordon Henderson
This is free software with ABSOLUTELY NO WARRANTY.
For details type: gpio -warranty

This Raspberry Pi is a revision 2 board.
pi@bananapi ~ $ gpio readall
+----------+-Rev2-+------+--------+------+-------+
| wiringPi | GPIO | Phys | Name   | Mode | Value |
+----------+------+------+--------+------+-------+
|      0   |  17  |  11  | GPIO 0 | ALT4 | Low   |
|      1   |  18  |  12  | GPIO 1 | ALT0 | High  |
|      2   |  27  |  13  | GPIO 2 | ALT4 | Low   |
|      3   |  22  |  15  | GPIO 3 | ALT4 | Low   |
|      4   |  23  |  16  | GPIO 4 | OUT  | High  |
|      5   |  24  |  18  | GPIO 5 | OUT  | Low   |
|      6   |  25  |  22  | GPIO 6 | ALT4 | Low   |
|      7   |   4  |   7  | GPIO 7 | IN   | Low   |
|      8   |   2  |   3  | SDA    | ALT5 | Low   |
|      9   |   3  |   5  | SCL    | ALT5 | Low   |
|     10   |   8  |  24  | CE0    | IN   | Low   |
|     11   |   7  |  26  | CE1    | IN   | Low   |
|     12   |  10  |  19  | MOSI   | IN   | Low   |
|     13   |   9  |  21  | MISO   | IN   | Low   |
|     14   |  11  |  23  | SCLK   | IN   | Low   |
|     15   |  14  |   8  | TxD    | ALT0 | Low   |
|     16   |  15  |  10  | RxD    | ALT0 | Low   |
|     17   |  28  |   3  | GPIO 8 | IN   | Low   |
|     18   |  29  |   4  | GPIO 9 | IN   | Low   |
|     19   |  30  |   5  | GPIO10 | OUT  | High  |
|     20   |  31  |   6  | GPIO11 | IN   | Low   |
+----------+------+------+--------+------+-------+
pi@bananapi ~ $
```

Figura 10: GPIO ReadAll

prende il nome di WiringBP.

Come utilizzare WiringPi sul Banana Pro / Pi

Prima di tutto bisogna scaricare WiringBP, digitando sul terminale:

Per Banana Pi:

```
git clone
https://github.com/LeMaker/Wi
ringBP -b bananapi
```

Per Banana Pro:

```
git clone
https://github.com/LeMaker/Wi
ringBP -b bananapro
```

Dopo il download, è necessario spostarsi nella directory WiringBP ed eseguire, sempre tramite terminale, quanto segue:

```
cd WiringBP/
sudo chmod +x ./build
```

Installiamo WiringBP:

```
sudo ./build
```

Finalmente è giunto il momento di controllare lo stato dei Pin GPIO tramite WiringPi, operazione che è

possibile effettuare digitando semplicemente sul terminale:

```
GPIO ReadAll
```

Apparirà una schermata simile a quella riportata in figura 10.

Programmazione in Python

Adesso che abbiamo tutto quello che ci serve per controllare i pin GPIO del nostro Banana Pi, non ci resta che creare il programma in Python che si occuperà di controllare lo stato del pulsante, avviare e spegnere il servizio Apache e segnalare lo stato tramite i led. Il programma da creare è mostrato nel Listato 1. Avviando questo programma appena creato, saremo in grado di impostare lo stato del servizio Apache.
Nel caso in cui voleste che il programma parta in automatico ad ogni avvio, dovrete impostarlo come "demone".

Conclusione

Quella che è stata presentata è un'ottima guida che permetterà a tutti di ottenere dei risultati fantastici e unici!
Personalmente non potrò fare a meno di utilizzare questo sistema, in quanto si presta a soddisfare ogni mia singola esigenza, rendendo la gestione dei miei documenti e file molto più comoda, compatta e (spero) più sicura.
Ancora una volta sono rimasto impressionato dalle potenzialità del Banana Pi, che non farà mai a meno di sorprendermi, in quanto l'unico limite d'applicazione dipende dalla propria fantasia.

Listato 1

```
import RPi.GPIO as GPIO
from time import sleep
import os

GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.IN) #PULSANTE
GPIO.setup(12, GPIO.OUT) #LED_VERDE
GPIO.setup(13, GPIO.OUT) #LED_ROSSO

state = 0

while True:
          input = GPIO.input(11)
          if (input == False):
          if (state == 1):
                   GPIO.output(12, True) #Accendo led verde
                        GPIO.output(13, False) #Spengo led rosso
                   print("Led green On")
                        os.system("sudo service apache2 start") #Avvio Apache
                   state = 0

             elif (state == 0):
                        GPIO.output(13, True) #Accendo led rosso
                   GPIO.output(12, False) #Spengo led verde
                   print("Led red On")
                        os.system("sudo service apache2 stop") #Spengo Apache
                   state = 1
             sleep(0.1)
```

Rimini Beach Mini Maker Faire®

14 e 15 NOVEMBRE
Rimini Fiera

ingresso ovest • ore 9/18

INNOVAZIONE, CREATIVITA' E DIVERTIMENTO
TRASFORMA LE TUE VISIONI IN REALTA'!

EXPO ELETTRONICA

STEAMPUNK COS-MAKER

FOTO MAKERS

MAKERS CALL MUTONIA

3D PRINTER

CRAFT

UAV WORLD

TALK, SPEECH, WORKSHOP

GAME PLAY ARENA MINECRAFT

MAK-ER FABLAB

MAKING MUSIC

KIDS & FAMILY

makerfairerimini.it

VALE COME RIDOTTO

ORGANIZZATO DA

PROMOTORI

PATROCINI

Lampada dimmerabile a comando vocale

di La Rosa Giuseppe

Questa lampada da tavolo si può controllare con comandi vocali e manualmente, è dotata di un dimmer azionabile tramite la voce e tramite un'apposita pulsantiera. Si può scegliere fra tre modi predefiniti d'illuminazione: leggere, studiare, dormire.

L'utilizzo dei parametri biometrici di un individuo per interfacciarsi a delle macchine è da sempre una meta che la tecnologia sta rendendo sempre più possibile. Lo scopo delle ricerche in questo settore dovrebbero permettere a breve di utilizzare le caratteristiche biometriche univoche di un individuo (impronte digitali, iride) in tutte le possibili applicazioni di rilevazione dell'identità: chiavi

Figura 1: Foto della lampada a comandi vocali

meccaniche, password, carta di credito e i modelli biometrici dinamici nelle applicazioni di riconoscimento delle espressioni del volto e del parlato. Quest'ultima applicazione ha compiuto nell'ultimo decennio molti passi in avanti. La tecnologia del riconoscimento del parlato, già in fase avanzata sugli Smartphone, si è, infatti, evoluta nel mercato Embedded, grazie ad un'applicazione trainante nell'utilizzare la voce per comandare piccoli dispositivi della vita quotidiana. In particolare, nel settore Embedded si sta distinguendo una notissima Casa produttrice: la Veear. L'EasyVR 3 è un piccolo modulo in grado di riconoscere il parlato e di parlare. Partendo da questo modulo abbiamo realizzato un comando vocale per il controllo di una lampada da tavolo a LED (vedi figura 1). Il dispositivo è in grado di riconoscere un comando composto da una parola e di invertire lo stato della propria uscita, attivando/disattivando o variando la luminosità di tre POWER LED. Le azioni che si possono applicare alla lampada sono l'attivazione, la disattivazione, l'alzare o l'abbassare la luminosità. E inoltre prevede tre modi predefiniti d'illuminazione: leggere, studiare, dormire. A questi tre modi sono stati associati altrettanti diversi livelli di luminosità, ad esempio al modo "dormire" è stato associato il livello luminoso minimo, allo "studiare" il livello luminoso più alto. Per interagire con la lampada, basta pronunciare queste parole: "Accendi" (attiva la lampada), "Spegni" (disattiva la lampada), "Alza" (Aumenta di uno Step la luminosità), "Abbassa" (diminuisci di uno Step la luminosità); poi, pronunciando le parole: "Dormire", "Studiare", "Leggere", la lampada eseguirà i livelli di luminosità predefiniti. Alla fine di ogni comando ricevuto, il modulo conferma con un suono. Nel nostro caso abbiamo usato queste parole

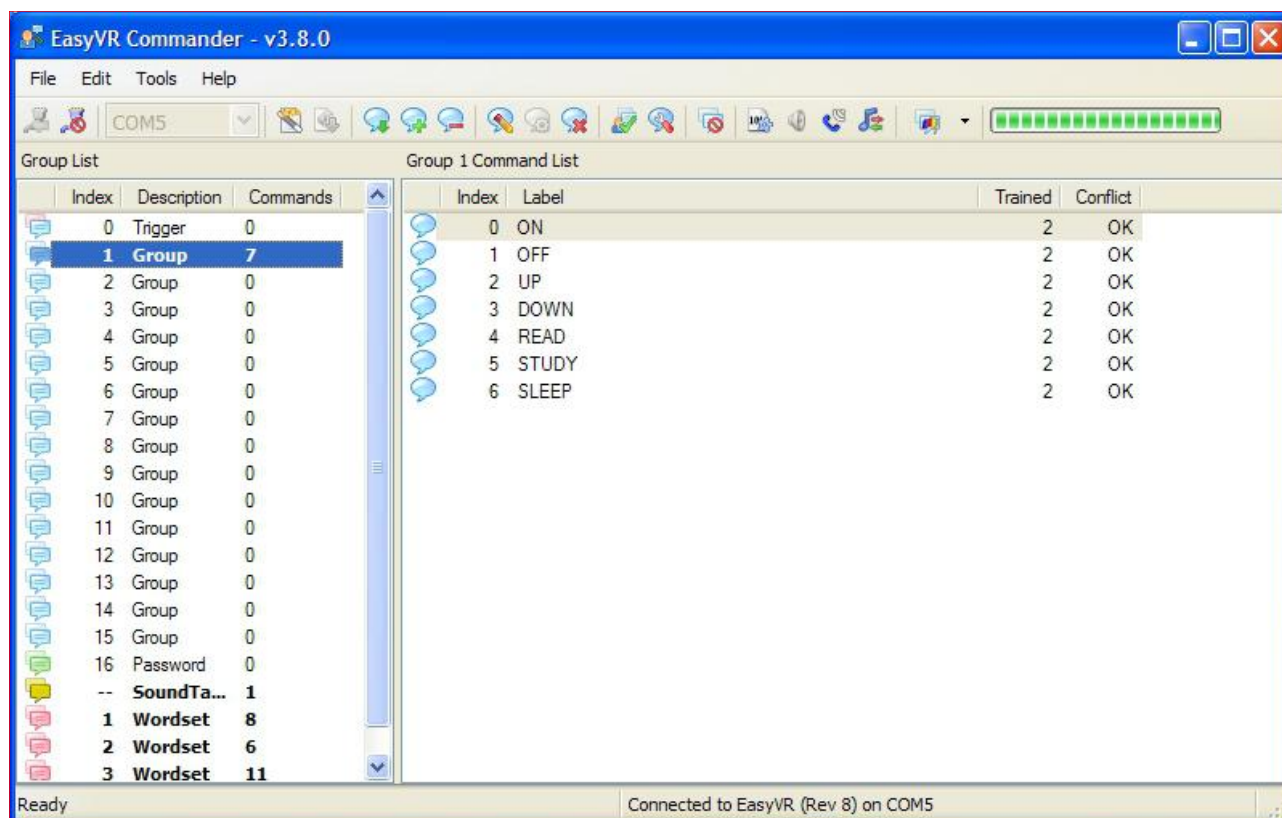



Figura 2: Schermata del software Easy Commander V3.8.

ma è possibile usare le locuzioni che ritenete più opportune. Oltre ai comandi vocali, è disponibile un cavo con una pulsantiera con tre pulsanti di cui uno serve per accendere e spegnere la lampada e gli altri due per regolare la luminosità.

Programmazione modulo EasyVR3

Qualsiasi dispositivo in grado di riconoscere il parlato può funzionare in base a due diverse metodologie, Speaker Independent, indipendente da chi parla o Speaker Dependent, dipendente da chi parla. La prima tecnologia consente di discriminare (riconoscere) una parola indipendentemente da chi la pronuncia: uomo, donna, bambino; la seconda consente di riconoscere una parola pronunciata, dalla stessa persona. Quest'ultima tecnologia, che tra l'altro è quella implementata nel nostro dispositivo, richiede una fase di apprendimento iniziale o Training. In pratica, la parola deve essere a priori appresa dal dispositivo per poter essere interpretata nel normale funzionamento. Per poter fare apprendere le parole precedentemente descritte bisogna usare il cavo USB di figura 3 ed innestare il connettore a sei poli nel connettore J7 del modulo EasyVR 3, collegare il cavo USB al PC. Aprite il programma Easy Commander V3.8 (Link per scaricalo alla fine dell'articolo). Una volta avviato ci troveremo di fronte ad un'interfaccia (vedi figura 2) davvero semplice e intuitiva. Infatti, per avviare la comunicazione basterà selezionare la porta seriale dove è presente il nostro modulo EasyVR 3 (nel nostro caso COM5) e poi cliccare sul pulsante "Connetti" (icona a sinistra "cavo seriale con freccetta"). Successivamente, per cambiare la lingua clicchiamo sull'ultima icona a destra e scegliamo italiano, questo passaggio servirà a migliorare il riconoscimento vocale. Ora spostiamoci sul Gruppo 1, ed inseriamo le Label cliccando sull'icona con il "fumetto e il simbolo + verde". Bisogna inserire le sette Label dall'indice 0 a 6 con gli stessi nomi riportati nella figura 2

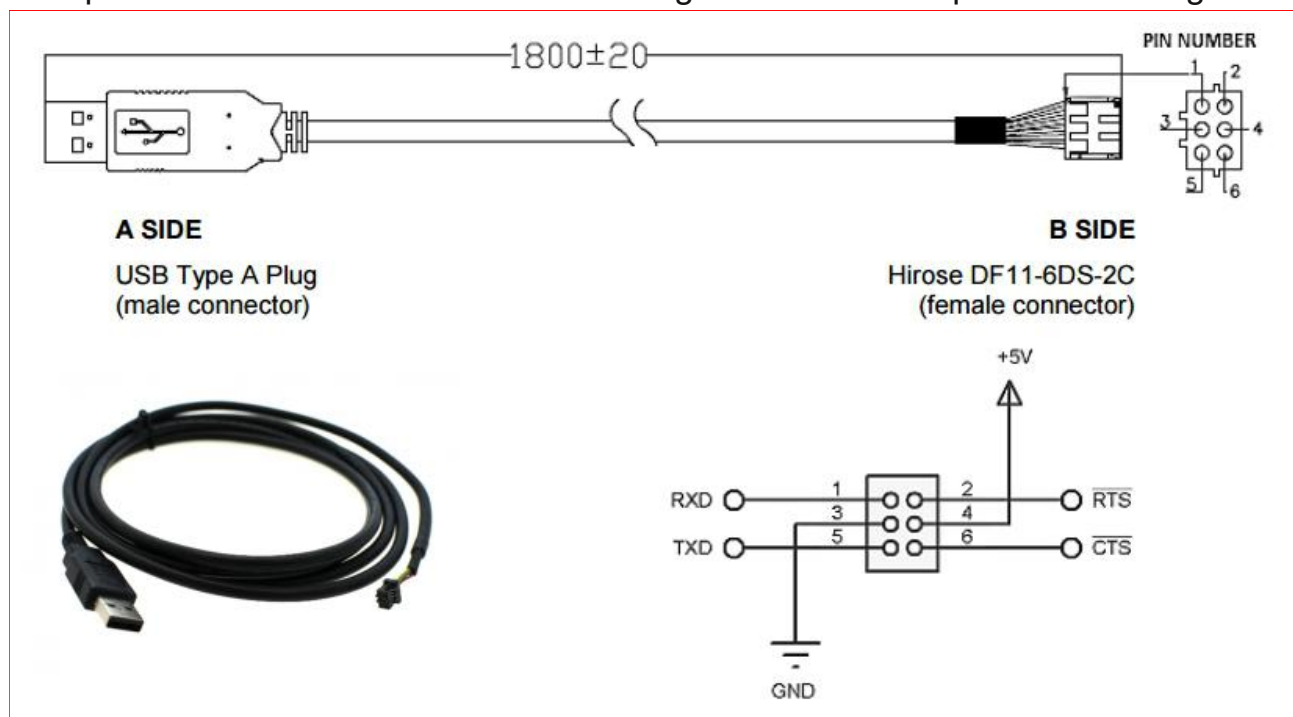
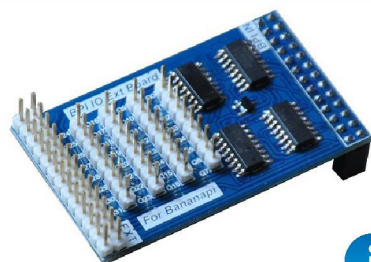



Figura 3: Cavo di programmazione per il modulo EasyVR 3 e pinout del connettore

cioè “ON” a “SLEEP” questo perché non useremo la funzione automatica che genera il codice per Arduino UNO, quindi tutto dovrà corrispondere alla figura 2. Create tutte le Label, dobbiamo far comprendere al nostro EasyVR 3 come si pronuncia quel comando per poi far sì che lo riconoscerà in futuro. Selezionando ogni Label e poi cliccando sull’icona “fumetto con l’ingranaggio”, ci apparirà una piccola finestra, che ci avvisa di pronunciare il comando da associare alla Label, dopo aver premuto il bottone “Phase 1”, entro un tempo massimo di 5 secondi. Una volta terminato di pronunciare il comando ci apparirà una seconda finestra che chiederà in sostanza la stessa cosa, ripetiamo l’operazione per la seconda volta ma questa volta vi consigliamo di spostarvi nella stanza o magari non utilizzare lo stesso e identico tono di voce. Questo serve a far aumentare le possibilità di riconoscimento vocale anche se vi trovate in punti diversi della stanza. Ripetete le stesse operazioni per ogni Label, pronunciando un comando diverso per ciascuna di esse. Una volta eseguiti questi semplici passaggi, potremo dare qualsiasi comando memorizzato al nostro EasyVR 3, senza uscire dal programma, potremo testare i comandi appena registrati tramite il bottoncino con l’icona con la “spunta verde”. Una volta finito il test, il modulo EasyVR 3 è pronto, basta cliccare sul bottoncino “Disconnetti” (vicino al bottoncino “Connetti”) e potete scollegarlo dal PC.

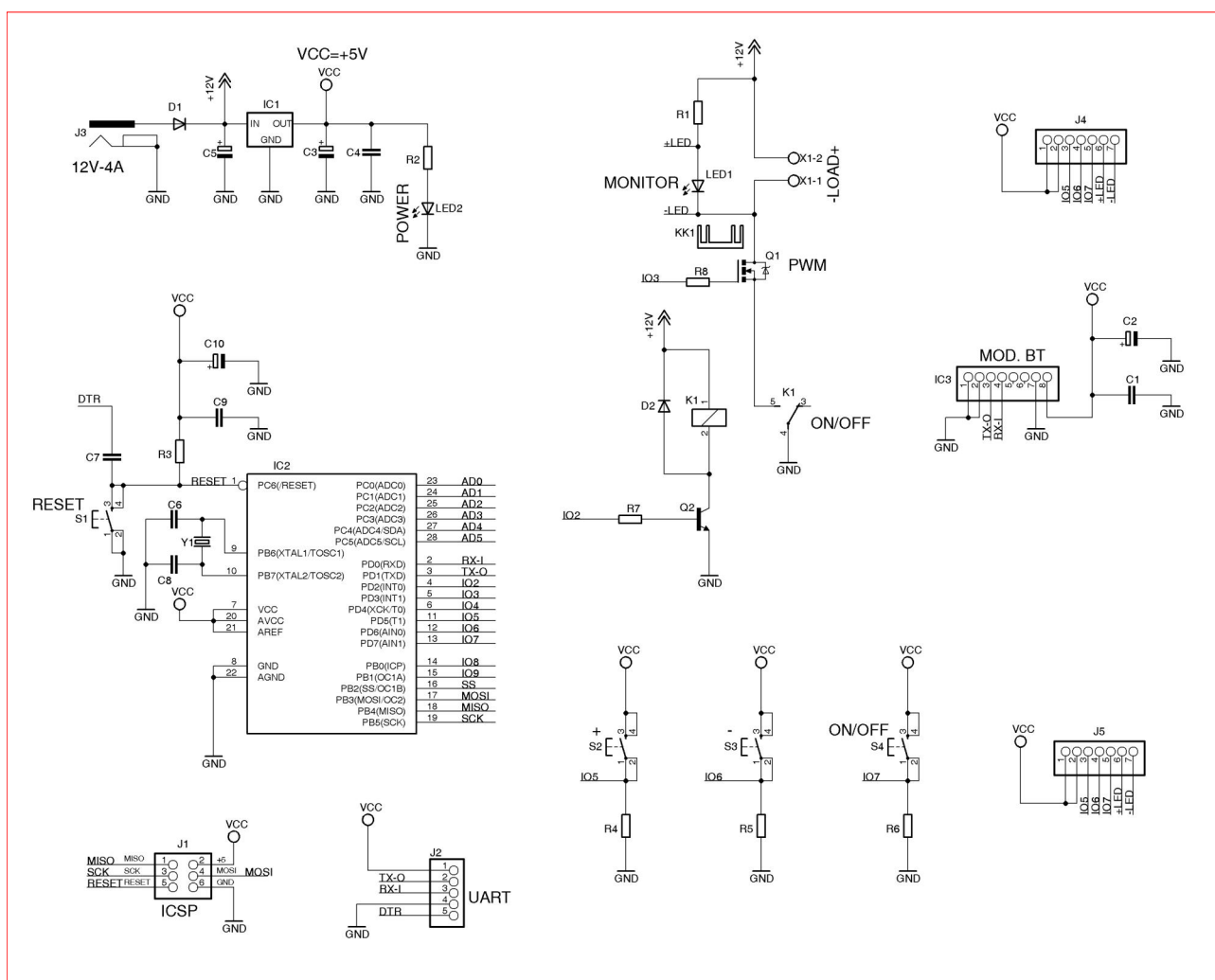



Figura 4: Schema elettrico della scheda dimmer a controllo vocale

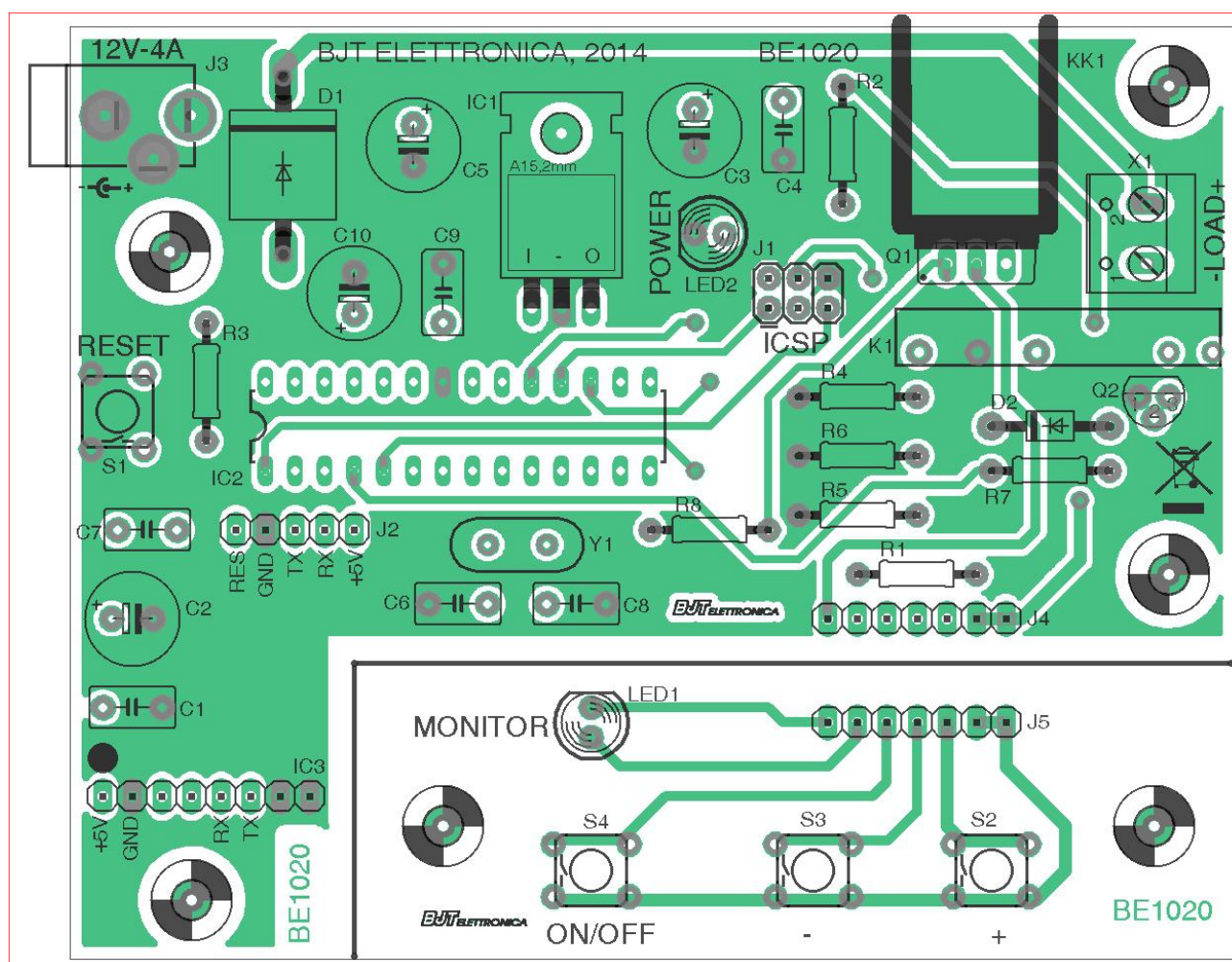



Figura 5: Piano di montaggio

Il circuito elettrico

In figura 4 è disegnato lo schema elettrico della scheda Dimmer a comandi vocali. Il relè K1 il Mosfet Q1 i tre pulsanti (S2, S3 e S4) vengono gestiti da Arduino UNO, un ATMEGA328P (IC2), ed il circuito che ne risulta è molto semplice, come si evince dalla figura 4. Il relè K1 ha il compito di spegnere e accendere i tre POWER LED da 1W (posti in serie, vedi figura 7) connessi al morsetto X1. Poiché il relè K1 non può essere connesso direttamente alla porta del microcontrollore IC2, perché ha un assorbimento elevato, maggiore della corrente massima erogabile dal microcontrollore, per ovviare a questo problema è stato usato il transistor Q2 in configurazione d'interruttore elettronico con il relativo diodo Damper D2, che agisce da soppressore di sovratensioni, impedendo alle extra tensioni generate dalla bobina del relè di attraversare il transistor Q2. Il Mosfet Q1 è pilotato dal segnale PWM generato dal microcontrollore IC2 (piedino IO3) e permette di pilotare grossi carichi, come i POWER LED. Il segnale PWM non è altro che una modulazione di larghezza d'impulso (o PWM, acronimo del corrispettivo inglese pulse-width modulation), è un tipo di modulazione digitale che permette di ottenere una tensione media variabile dipendente dal rapporto tra la durata dell' impulso positivo e di quello negativo, con questa tecnica si ottiene la variazione di luminosità che ci serve per i nostri ambienti da illuminare. Per collegare un MOSFET direttamente al

microcontrollore occorre scegliere un componente, definito Mosfet Logic Level, ovvero con tensione Vgs (tensione tra gate e source) inferiore ai 5 V. Questo è assolutamente indispensabile se vogliamo comandare il gate del Mosfet direttamente dal piedino IO3 del microcontrollore IC2. Il piedino IO3 del microcontrollore IC2 ha sufficiente capacità di corrente per portare in conduzione in modo ragionevole il Mosfet. Ma, come abbiamo detto, il gate richiede una tensione minima per ottenere la migliore resistenza in conduzione se la tensione applicata non è sufficiente, il Mosfet entra in conduzione, ma con una resistenza molto più elevata del minimo, e questo determina un inutile e indesiderato riscaldamento del componente. Per buona parte dei Mosfet la tensione di gate si aggira attorno a valori tra 8V e 12 V. Mentre la logica del microcontrollore è alimentata a 5 V. Ne deriva che occorrerà un Mosfet la cui tensione di gate sia inferiore a 5 V per la conduzione. I dispositivi Logic gate hanno appunto Vgs attorno ai 4,2 V e quindi sono quelli adatti a essere comandati direttamente da un'uscita a livello logico 5V. Il Mosfet Q1 scelto per questa applicazione è un BUK9535 Logic Level, per approfondimenti leggere il Datasheet. I tre pulsanti S2, S3, e S4 hanno diverse funzioni: S4 di accendere e spegnere i POWER LED, gli altri due per aumentare (S2) e per diminuire (S3) la luminosità dei POWER LED. Essi sono collegati in pull-down tramite le resistenze R4, R5 e R6, collegate verso la massa (pull-down), il potenziale ai piedini d'ingresso del microcontrollore (piedini IO5, IO6 e IO7) con il pulsante non premuto è stabilito dalla GND (massa). Con le resistenze di pull down abbiamo assicurato un determinato livello di tensione per entrambe le posizioni dei pulsanti. Il connettore J1 ha due funzioni: una per la programmazione "In Circuit" ed è

Elenco componenti

R1	Non usato
R2	470 Ω 1/4 W
R3÷R6	10 KΩ 1/4 W
R7	12 KΩ 1/4 W
R8	330 Ω 1/4 W
C1	Non usato
C2	Non usato
C3	100 μF 35 V elettrolitico
C4	100 nF poliestere
C5	100 μF 35 V elettrolitico
C6	22 pF ceramico
C7	100 nF poliestere
C8	22 pF ceramico
C9	100 nF poliestere
C10	100 μF 35 V elettrolitico
D1	6A60 diodo
D2	1N4007 diodo
Q1	BUK9535
Q2	BC337
IC1	L7805CV
IC2	ATMEGA328P
IC3	Non usato
LED1	Non usato
LED2	LED 5 mm verde
Y1	Quarzo 16 MHz
K1	Relè 12 V/6 A
S1÷S4	Pulsante c.s.
J1	STRIP maschio 3+3 pin
J2	STRIP maschio 5 pin
J3	Presa DC 90° 5,5x2,1 mm
J4÷J5	Terminale a saldare
X1	Morsetto 2 poli
KK1	Non usato
N.1	Zoccolo 14+14 pin
N.1	Vite 3x15 mm più dadi
N.1	Cavo piatto tel. 4 poli 1 m
N.3	Power Led 1 W

Figura 6: A sinistra master lato saldature a destra master lato componenti

utile per caricare il Bootloader di Arduino UNO, l'altra per connettere il microcontrollore al modulo EasyVR 3. Il connettore J2 permette di caricare il firmware dall'IDE di Arduino UNO tramite convertitore seriale USB/TTL. La scheda deve essere alimentata con un alimentatore stabilizzato da 12V, in grado di erogare una corrente di 3,5 A, corrente sufficiente per alimentare i tre POWER LED da 1 W.

Il firmware

Il Firmware dell'ATMEGA328P è stato scritto in linguaggio C (Arduino UNO). Il firmware riconosce la sequenza di caratteri trasmessi dal modulo EasyVR 3 tramite l'interfaccia seriale UART implementata via Software sui pin 18 e 19 (corrispondenti ai pin 12 e 13 di Arduino UNO) dell'ATMEGA328P, connettore J1. Il Firmware, tramite la porta Hardware sul connettore J2, trasmette tutti i messaggi di errore e di conferma dei comandi, basta aprire il Serial Monitor di Arduino per verificare tutte le operazioni eseguite dal modulo EasyVR 3. Alla riga 198 del sorgente del firmware è stato inserito un ciclo "do…while" che permette di utilizzare i tre pulsanti in attesa dei comandi vocali ottenendo sia funzionamento manuale e vocale. Non dimenticate d'installare la libreria EasyVR (Link alla fine dell'articolo) nell'IDE di Arduino per poter programmare la scheda ed eseguire le vostre modifiche.

Realizzazione delle schede e collaudo

Passiamo adesso alla costruzione della scheda che si presenta abbastanza semplice, la basetta è del tipo doppia faccia con fori metallizzati e si prepara sulla base dalle tracce di figura 6. Ottenuto il circuito stampato, iniziate a montare la scheda (seguendo il piano di montaggio di figura 5) e i componenti richiesti dall'"Elenco componenti". Inserite le resistenze, in seguito il diodo D2, per ultimo il

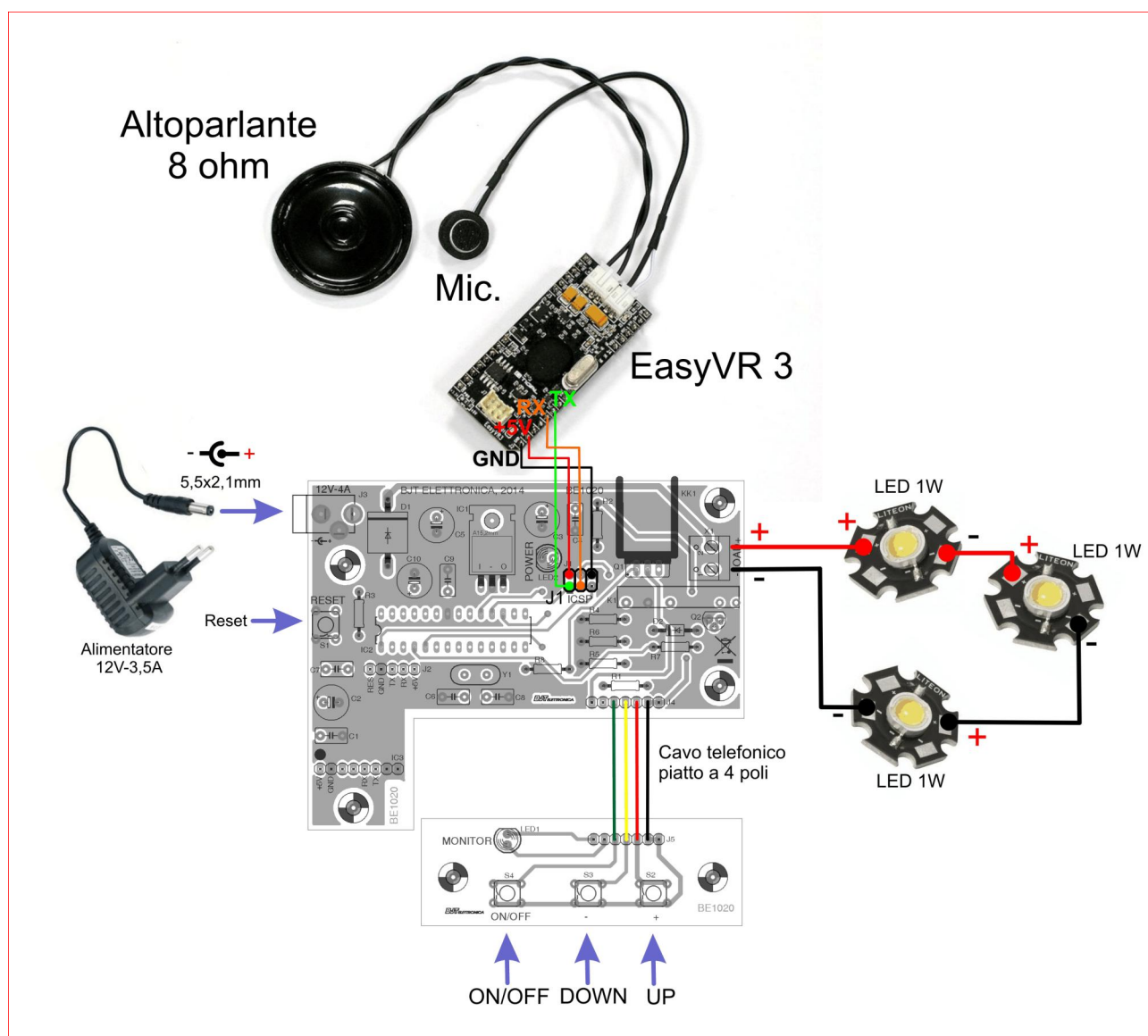



Figura 7: Schema di collegamento

diodo D1 che va saldato a 5 mm dalla piastra. Saldate lo zoccolo per l'integrato IC2, proseguite con i condensatori non polarizzati e poi gli elettrolitici, il quarzo Y1, gli STRIP J2 e J1, i pulsanti, il LED, e i connettori X1, J3 e per ultimi il relè e il Mosfet Q1. Se si dispone di un ATMEGA328P già con Bootloader caricato si può passare al caricamento del firmware tramite un convertitore USB/TTL e collegarlo al connettore J2 presente sulla scheda. Altrimenti prima bisogna caricare tramite un programmatore il Bootloader dal connettore J1. Eseguite il collegamento rappresentato in figura 7, connettete i tre POWER LED in serie e poi al morsetto X1, connettete infine il modulo EasyVR 3 al connettore J1. La parte del circuito stampato con i pulsanti può essere tagliata e staccata dalla scheda per poi essere connessa con del cavo piatto telefonico a 4 poli alla scheda Dimmer ed in seguito rivestita con della guaina termoretraibile del diametro di 33 mm, come in figura 8. Finite le operazioni di cablaggio e di programmazione del modulo EasyVR 3 (descritte in precedenza), collegate un alimentatore a 12V in grado di erogare 3,5A. A que¬sto punto potete collaudare la scheda: premendo il pulsante S4 si accenderanno i tre POWER LED alla minima luminosità e con i due

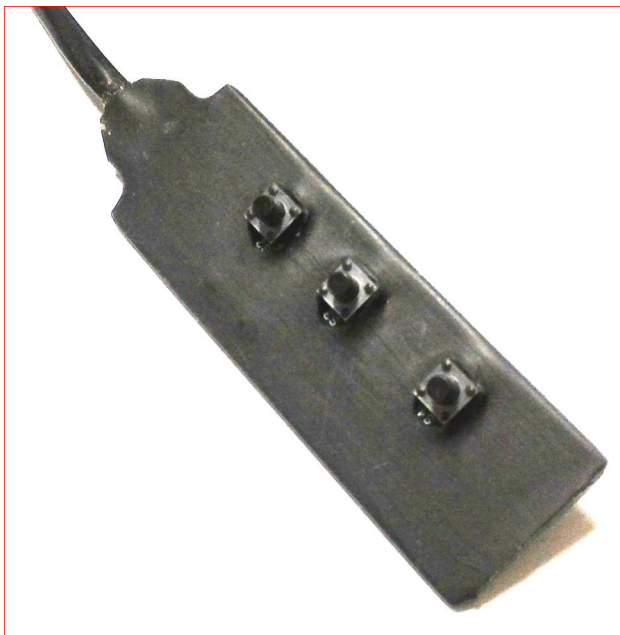



Figura 8: Pulsantiera per il controllo manuale

pulsanti S2 e S3 si potrà aumentare e diminuire la luminosità.
Impartite tutti i vostri comandi vocali che avete programmato in precedenza verificando che vengano riconosciuti senza troppe ripetizioni, se così non fosse consigliamo di ripetere la programmazione del modulo, variando nelle due fasi di acquisizione il tono della voce.
Il contenitore si può costruire (vedi figura 1) oppure si può utilizzare qualsiasi lampada che trovate nei centri "fai da te" e operando le opportune modifiche per l'alloggiamento dei POWER LED e della scheda del relativo microfono e altoparlante. Lasciamo la personalizzazione del contenitore al lettore che avrà vari modi di realizzare questa lampada da tavolo.

Conclusione

Tutti i file per la realizzazione, come già è stato detto, sono disponibili ai Link indicati qui sotto.
Come avete letto in precedenza la lampada può essere personalizza sia a livello di Design che di Firmware, quindi adattabile alle vostre esigenze e gusti.
Di seguito trovate tutti i Link per scaricare sia file per la realizzazione del progetto, ma anche i Link dove trovare il materiale per la realizzazione.

Links

Per scaricare il Software Easy Commander V3.8 e la libreria per Arduino UNO:

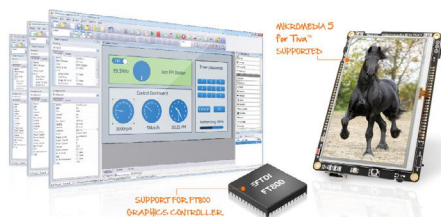
http://www.veear.eu/downloads/

Se volete acquistare un modulo EasyVR 3:

http://www.elettroshop.com/multi-language-speech-recognition-module-with-serial/

Per scaricare i sorgente per l'ATMEGA328P:

http://larosagiuseppe.altervista.org/FWEasyVRDimmer.rar

Per guardare il video del progetto in funzione:

https://www.youtube.com/watch?v=en81GvjNLlA

Arduino e l'orologio atomico

di Girolamo D'Orio

Ecco come realizzare un orologio-datario con visualizzazione su LCD. Il modulo RTC è aggiornato una volta al giorno, tramite la ricezione del segnale radio ad onde lunghe DCF77.

Nel precedente articolo ho illustrato come raggiungere una soddisfacente precisione della misura del tempo, regolando il modulo RTC con due metodi: hardware e software. Questa volta, ho cercato di aggirare il problema della precisione ricorrendo alla ricezione del segnale a onde lunghe in cui è trasmesso l'orario assoluto scandito da tre orologi atomici. Il segnale in questione è nominato DCF77, un sistema tedesco in cui è trasmesso l'orario corrente e le previsioni meteo. In questo articolo parlerò solamente della parte che riguarda l'orario. L'ora assoluta è scandita dai tre orologi atomici, due al Cesio e uno al Rubidio. Il loro compito è di generare una portante con una frequenza che abbia un margine

Figura 1: Copertura del segnale DCF77

Figura 2: Le antenne che trasmettono il segnale

di errore estremamente basso. Il margine di errore è stimato a circa 2*10^-13 sulla media di cento giorni. Le antenne gigantesche situate a Mainfligen, vicino a Francoforte, trasmettono il segnale ad una frequenza di 77,5 KHz. I due trasmettitori da 50KW ciascuno, trasmettono in parallelo per garantire continuamente il segnale emesso. Sfruttano la ionosfera per cercare di avere più copertura possibile in Europa. E' stimato che la copertura del segnale si aggira tra i 1900 e i 2100 km. L'unico neo di questo segnale è la sua "lunghezza". I tanti bit che lo compongono ci portano l'informazioni necessarie, il segnale va campionato nello spazio temporale di 60 secondi. Al sessantesimo secondo volutamente non è trasmesso nulla, proprio per farci capire che al prossimo secondo incomincia la nuova trasmissione. Dato che è impossibile ricevere senza disturbi questo segnale in modo continuativo, mi appoggio ad un modulo RTC che continuerà a

Figura 3: Formato dei dati

tenere il conto del tempo anche quando il segnale viene a mancare. Guardando lo schema si nota che fa da padrona la comunicazione I2C. L'integrato PCF857N, un I/O expander, viene utilizzato per pilotare l'LCD sfruttando appunto la comunicazione I2C. Anche il modulo RTC sfrutta tale protocollo, mentre il modulo adibito alla ricezione del segnale DCF77 va collegato all'interrupt 0 del microcontrollore, con la resistenza di pull-up. Noterete inoltre che è presente un notevole numero di condensatori da 100 nF. Li ho posizionati nei pressi di ogni singola alimentazione di moduli e integrati, al fine di cercare di diminuire il più possibile le auto-oscillazioni; esse potrebbero, infatti, disturbare il ricevitore DCF77 e il microcontrollore.

Descrizione del circuito Hardware

Il circuito ha un assorbimento modesto, qualsiasi piccolo trasformatore può andare certamente bene, l'importante che il secondario abbia una tensione di uscita compresa tra 8V e 15V. Nel caso usiate un trasformatore con un secondario superiore a 12V consiglio di installare una piccola aletta di dissipamento al

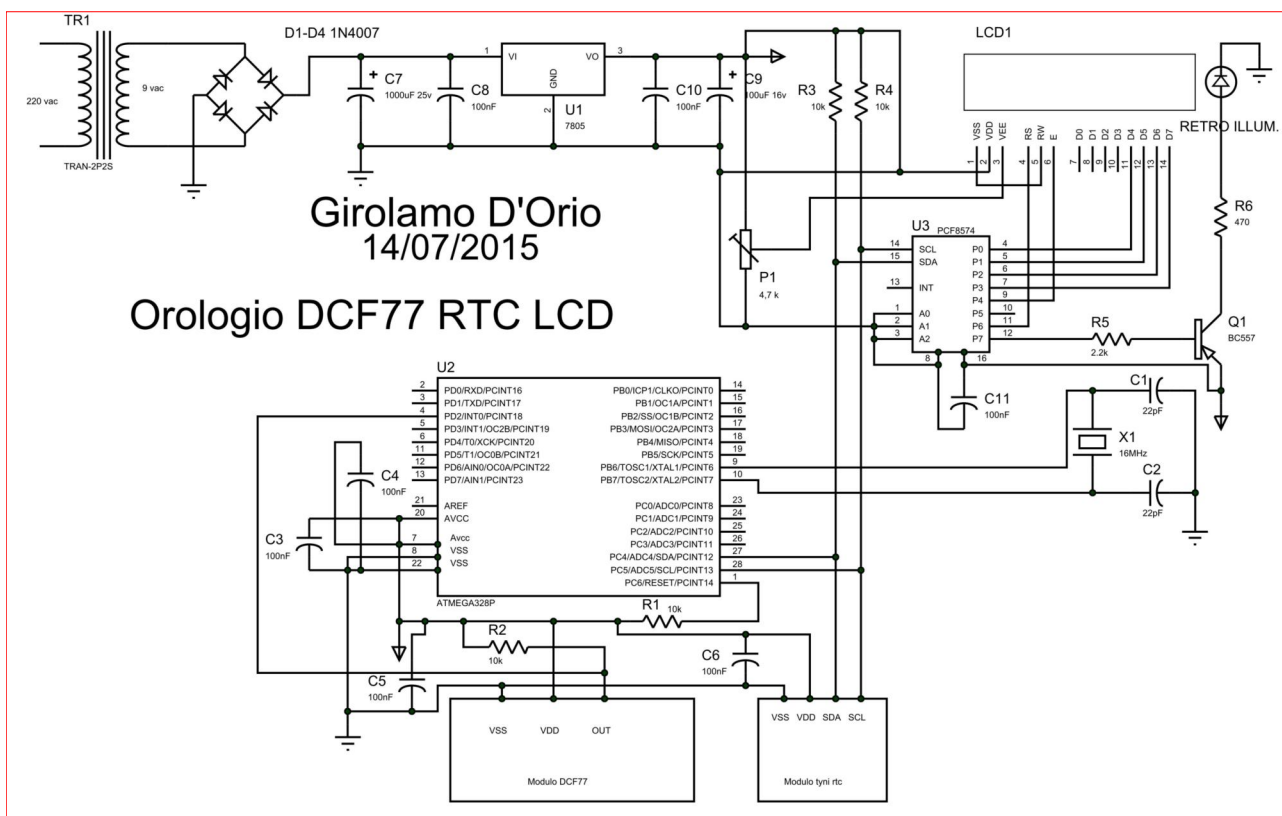



Figura 4: Schema elettrico

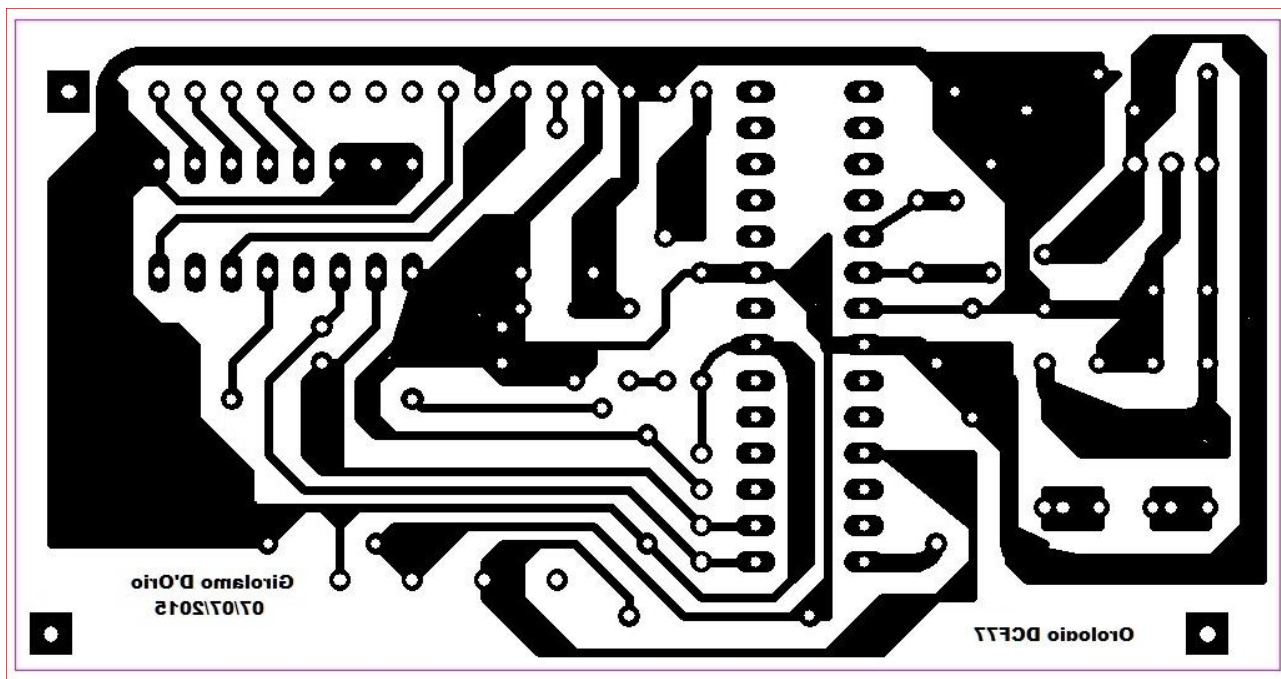



Figura 5: Il master del PCB (88,6 x 45,6 mm)

regolatore di tensione 7805. Non occorre necessariamente alimentare il circuito con un piccolo alimentatore stabilizzato; nel circuito è già presente lo stabilizzatore. Il circuito, dal punto di vista elettronico, è molto semplice: Occorre prestare attenzione all'integrato PCF857N: esso non deve essere erroneamente confuso con il PCF8574A o il PCF8574AN, in quanto si differenziano solamente per l'indirizzo I2C. Eventualmente occorre modificare solamente l'indirizzo nel sorgente dopo aver consultato il Datasheets. Altro errore potenziale è nel display LCD. E' il classico HD44780 ma si differenzia da quelli più comuni, in quanto i pin 1 e 2 fanno parte del Led per la retroilluminazione.

Descrizione del software per Arduino

Il sorgente è ben commentato e

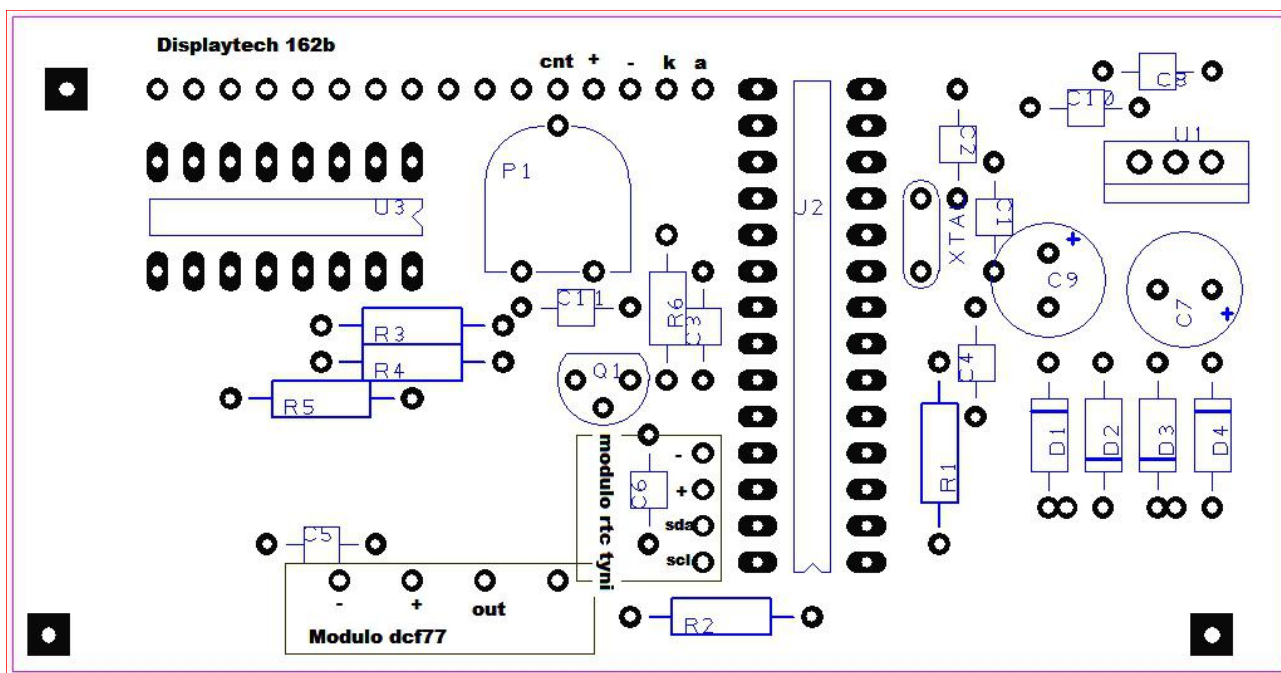



Figura 6: Montaggio componenti

Figura 7: Il modulo RTC

non troverete nessuna difficoltà nel comprenderlo. Per prima cosa descrivo le librerie che ho utilizzato.

RTClib
La RTClib ci permette di ricavare dal modulo omonimo l'ora e la data.

Funkur
La libreria Funkur, il cuore del sorgente, permette di decriptare il segnale radio ricevuto dal modulo DCF77. Un ringraziamento al creatore Matthias Dalheimer.

LiquidCrystal_I2C e Wire
Le due librerie hanno il compieto di pilotare l'LCD in I2C.

WATCHDOG
La libreria WATCHDOG è già presente nel compilatore IDE.

Per la prima volta mi sono avventurato nella creazione di caratteri speciali da inviare all'LCD, per permettere di capire quando siamo in presenza di un segnale valido. Ho provato a replicare il simbolo dell'antenna che, quando "doppia", ci fa capire che siamo in presenza di un

Elenco componenti

R1-R4	10 K Ω ¼ W
R5	2,2 KΩ ¼ W
R6	470 Ω ¼ W
P1	4,7 K Ω trimmer
C7	1000 uF 24V elettr.
C9	100 uF 16V elettr.
C1,C2	22 pF poliestere
C3-C6, C8,C10	100 nF poliestere
U1	7805
U2	ATmega328p-pu
U3	PCF8574N
D1-D4	1N4007
Q1	BC557 PNP
XTAL	16 Mhz
LCD	Displaytech 162-B
TR1	Trasfor. 0-9v 300mA

segnale valido. Per permettere di decriptare i dati in modo più efficiente e sicuro, ho impostato il sorgente in modo che il microcontrollore non si deve occupare di fare altro sino a quando esso non ha estrapolato data e ora corrente. Questo perché? Se provate direttamente a far scrivere in LCD il conteggio del tempo mentre ancora non abbiamo ricevuto un segnale valido, si potrebbero presentare errori e addirittura la campionatura spesso non andrebbe a buon fine. Ciò avviene

Figura 8: Il modulo Ricevitore DCF77

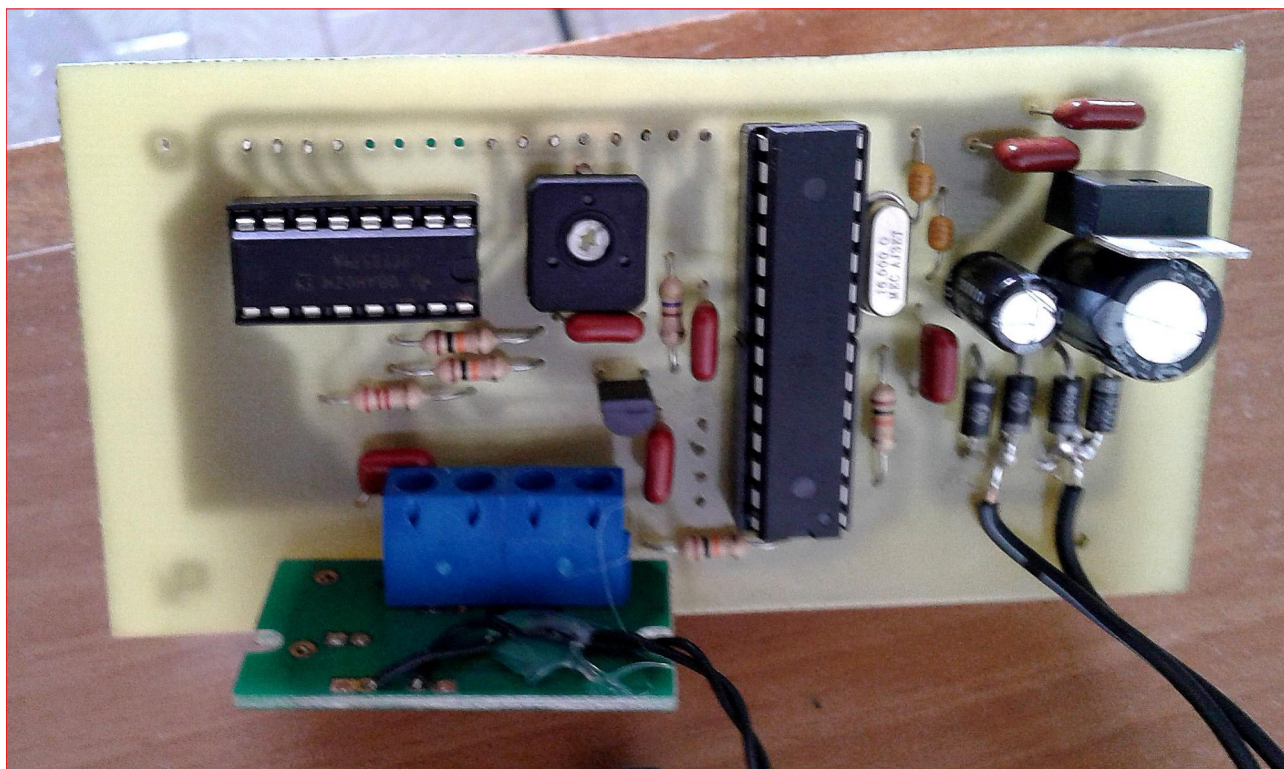



Figura 9: Il lato componenti

perché Il microcontrollore perde diversi millisecondi a trascrivere su LCD, quindi il timer interno, che si occupa di ricevere il segnale e che associa i bit ai secondi, è in qualche modo disturbato. Ricevuto il segnale valido, analizzo il dato relativo alla variabile dell'anno. Se risulta maggiore di 0, andrò ad aggiornare il modulo RTC. A questo punto il programma tralascia la parte di ricezione del segnale e si occupa di visualizzare su LCD la data e l'ora provenienti dal modulo RTC. La variabile di comodo nominata "aggiornamento" cambia il suo

Figura 10: Particolare del montaggio LCD

stato, quindi al prossimo ciclo del programma, il modulo RTC non verrà aggiornato ancora, in quanto se perdiamo la ricezione scomparirebbe l'ora e data su

Figura 11: Montaggio modulo RTC sul lato piste

LCD. Il modulo RTC potrà avere errori di qualche secondo al massimo nell'arco delle 24 ore. Per correggere questo errore ho deciso di sperimentare il WATCHDOG, che ogni 24 ore resetta il microcontrollore. In questo modo almeno una volta al

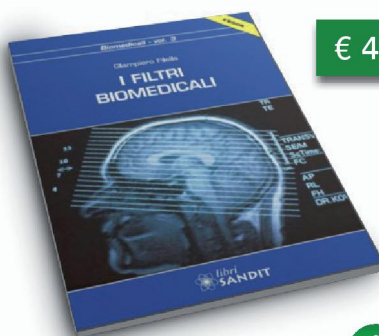
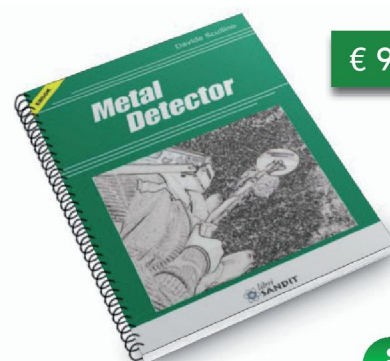
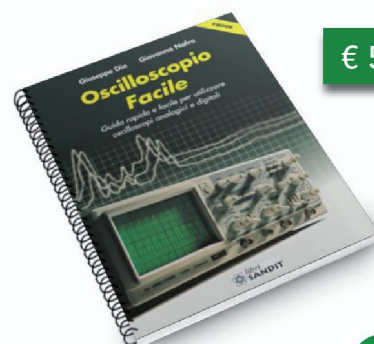
Figura 12: L'orologio in funzione

giorno il modulo RTC verrà aggiornato nuovamente avendo cosi la certezza di avere un orario corretto. Il sorgente prevede di far svolgere questa operazione alle 23:58pm. Quindi se avete esigenze particolari basta correggere tale comportamento. Personalmente consiglio di scegliere un orario notturno, poiché la ricezione è senza dubbio migliore e meno disturbata. Fino a che la ricezione non va a buon fine, sull'LCD compare la scritta "IN RICEZIONE". Consiglio di tenere il dispositivo lontano da apparecchiature elettriche ed elettroniche di qualsiasi tipo, perché esse tendono a disturbare il segnale. La migliore ricezione avviene di notte, quando molte trasmissioni radio sono disattivate. Il link del video su Youtube mostra la ricezione del segnale anche di giorno e in condizioni non ideali.

Link Video:
https://www.youtube.com/watch?v=b1bTn5NMxz0

Link libreria FunkuHR:
https://github.com/fiendie/Funkuhr

Buon divertimento e buona realizzazione a tutti.

Figura 13: L'orologio in ricezione

Listato completo

```
#include "Funkuhr.h"
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
#include "RTClib.h"
#include <avr/wdt.h> // libreria watchdog
RTC_DS1307 rtc;
int aggiornamento; // variabile di comodo
LiquidCrystal_I2C lcd(0x20,16,2);  // set the LCD address to 0x20 for a 16 chars and 2
line display
byte newChar[8] = {   //creazione carattere speciale
        B10001,  // Simbolo di antenna per segnalare
        B10001, // che la ricezione ancora non è valida
        B10001,
        B10001,
        B01010,
        B00100,
        B00100,
        B00100
};
byte newChar2[8] = {  // creazione carattere speciale
        B11011,  // simbolo di antenna "doppia" per segnalare
        B11011,  // la ricezione di un segnale valido
        B11011,
        B10101,
        B01010,
        B00100,
        B00100,
        B00100
   };
Funkuhr dcf(0, 2, 13, false);
struct Dcf77Time dt = { 0 };
uint8_t curSec;

void dumpTime(void)  // routine presa dall'esempio della libreria
{
           Serial.println("DCF77 Time");
           // Print date
           Serial.print(" ");
           if(dt.day < 10)
                      Serial.print("0");
           Serial.print(dt.day, DEC);
           Serial.print(".");
           if(dt.month < 10)
                      Serial.print("0");
           Serial.print(dt.month, DEC);
           Serial.print(".");

           if(dt.year == 0)
           {
                      Serial.print("000");
           }
           else
           {
                      Serial.print("20");
           }

           Serial.print(dt.year, DEC);

           if(dcf.synced())
           {
                      Serial.println(" ");
                      Serial.print(" ");
```

```
            }
            else
            {
                        Serial.println(" ");
                        Serial.print("~");
Serial.print("in ricezione...");
            }
            // Print Time
            if (dt.hour < 10)
                        Serial.print("0");
            Serial.print(dt.hour, DEC);
            Serial.print(":");

            if (dt.min < 10)
                        Serial.print("0");

            Serial.print(dt.min, DEC);
            Serial.print(":");

            if (dt.sec < 10)
                        Serial.print("0");

            Serial.println(dt.sec, DEC);

            Serial.println(" ");
}

void setup(void)
{
   wdt_disable();  //disabilito il WatchDog, consigliato in quanto
   // la realizzazione è in STAND-ALOne
   aggiornamento=0; //variabile di comodo quando è a 0 RTC
                        // è pronto a ricevere l'aggiornamento
 lcd.init();      // inizializzazione  lcd
 lcd.backlight(); // accesione retroilluminazione LCD
 lcd.createChar(0, newChar); //creazione caratteri
 lcd.createChar(1, newChar2);
 lcd.setCursor(0,0);
 lcd.write(0); //scrive in lcd il carattere associato allo 0
 lcd.setCursor(4,1);
 lcd.print("in ricezione");
  Serial.begin(9600);
            dcf.init();
#ifdef AVR
  Wire.begin();
#else
  Wire1.begin(); // Shield I2C pins connect to alt I2C bus on Arduino Due
#endif
  rtc.begin();
  if (! rtc.isrunning()) {
    Serial.println("RTC is NOT running!");
 }
}

void loop(void)
{
while (dt.year ==0){ // condizione che permette di eseguire solamente
// la routine per la ricerca del segnale
  dcf.getTime(dt);
            if(dt.sec != curSec)
            {
                        dumpTime();
            }
            curSec = dt.sec;
```

```
if (dt.year >0){ // avvenuta la ricezione dato che
// la variabile assume un valore >0 usciamo dalla condizione precedente
break;
}
}
if (dt.year >0){  // condizione per aggiornare il modulo RTC
  DateTime now = rtc.now();
  if (aggiornamento==0){
   rtc.adjust(DateTime(2000+dt.year, dt.month, dt.day, dt.hour, dt.min,dt.sec ));
   aggiornamento=1;
   dt.year=0;
   }
   // stampiamo su LCD ORA e DATA fornite dal modulo RTC
  lcd.clear();
lcd.setCursor(4,0);
lcd.print(now.hour(), DEC);
lcd.print(":");
lcd.print(now.minute(), DEC);
lcd.print(":");
lcd.print(now.second(), DEC);
lcd.setCursor(4,1);
lcd.print(now.day(), DEC);
lcd.print("/");
lcd.print(now.month(), DEC);
lcd.print("/");//lcd.print("20");
lcd.print(now.year(), DEC);
// Test di verifica in fae di prova
/*
Serial.println("");
Serial.println(dt.year);
Serial.println("");
// test
Serial.print("rtc orarario");
Serial.println(" ");
  Serial.print(now.year(), DEC);
    Serial.print('/');
    Serial.print(now.month(), DEC);
    Serial.print('/');
    Serial.print(now.day(), DEC);
    Serial.print(' ');
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    Serial.print(':');
    Serial.print(now.second(), DEC);
    Serial.println();
*/
lcd.setCursor(0,0);
lcd.write(1); // scrittura del carattere speciale che
// conferma segnale radio valido
 // Condizione per resettare il micro ogni 24 ore
   if (now.hour()==23 && now.minute()==58 && now.second()<2){
  aggiornamento=0;
  wdt_enable(WDTO_250MS); //abilito il WATCHDOG
  }
}
// se il segnale non è valido compare il simbolo di antenna "vuota"
else{
lcd.setCursor(0,0);
lcd.write(0);
}
delay(1000);
}
```

DIRETTORE RESPONSABILE
Antonio Cirella

DIRETTORE TECNICO
Giovanni Di Maria

Hanno collaborato in questo numero:
Vincenzo Sorce, Davide Fiorino, Ivan Scordato, Girolamo D'Orio, Giuseppe La Rosa.

Direzione Redazione
INWARE srl
Via Giotto, 7 - 20032 Cormano (MI)
Tel. 02.66504794 -Fax 02.42101817
info@inwaredizioni.it - www.inwaredizioni.it

Redazione:
fe@inwaredizioni.it

Pubblicitá per l'Italia
Agostino Simone
Tel. 347 2230684
media@inwaredizioni.it

Europe and Americas
Elisabetta Rossi
Tel. +39 328 3245956
international@inwaredizioni.it

Asia
Cybermedia Communications Inc.
asia@inwaredizioni.it

Rest of the world
Inware Edizioni srl
Tel. +39 02 66504794
info@inwaredizioni.it

Ufficio Abbonamenti
INWARE srl
Via Giotto, 7 - 20032 Cormano (MI)
Per informazioni, sottoscrizione
o rinnovo dell'abbonamento:
abbonamenti@inwaredizioni.it
Tel. 02.66504794 - Fax 02. 42101817
L'ufficio abbonamenti è disponibile
telefonicamente dal lunedì al venerdì
dalle 14,30 alle 17,30.

Autorizzazione alla pubblicazione
Tribunale di Milano n. 647 del 17/11/2003